

# **Research on Automatic Classifiers Techniques For Text Classification**

**by**

**Mahmoud Ibrahim Elhosiny Elmarhoumy**

A Doctor dissertation submitted to  
the Graduate School of Systems Innovation Engineering  
at The University of Tokushima for the Degree of  
**Doctor of Engineering**

In

**Information Science and Systems Engineering**



**March 2014**

Department of Information Science and Intelligent Systems  
The University of Tokushima, Tokushima  
8506-770Japan

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

﴿ اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ ﴿١﴾ خَلَقَ الْإِنْسَانَ مِنْ عَلَقٍ ﴿٢﴾ اقْرَأْ وَرَبُّكَ الْأَكْرَمُ ﴿٣﴾

الَّذِي عَلَّمَ بِالْقَلَمِ ﴿٤﴾ عَلَّمَ الْإِنْسَانَ مَا لَمْ يَعْلَمْ ﴿٥﴾

صدق الله العظيم

*“Read. Read in the name of thy Lord who created; [He] created the human being from blood clot. Read in the name of thy Lord who taught by pen: [He] taught the human being what he did not know.”*

*-Al-Qur'an (96:1-5)*

## Acknowledgment

First, I am very much grateful to my Creator Almighty **ALLAH**, the most Merciful and beneficent, for giving me the strength, patience and opportunity to work on my research field during the course of my study at the University of Tokushima and all my life.

I would like to express my thanks to the **Tokushima foreigner affair section** for giving me this opportunity to study my master courses at Tokushima University, Japan, and supporting partially the work described in this thesis. My deepest thanks are to them for their continuous support throughout the whole period of my studying in Japan.

My deep thanks come to my supervisor **Professor Fuji Ren**, for his guidance, kind support, help, and continuous encouragement, and for his patience hearing to my problems throughout the accomplishment of these studies. I appreciate his in-exhaustive efforts, unending cooperation and advice, his deep insights that always found solutions when problems supervened and very creative criticism. I also thank him for introducing me to the Natural Language Processing fields of Information science and intelligent system department. I also wish to thank **Professor Kenji Terada** and **Professor Masami Shishibori**, for accepting to be staff members of the dissertation committee of this thesis and also for the time they spent in reading, revising and verifying the thesis.

My thanks and appreciation to all my colleagues and friends at the Department of Information Science and Intelligent System, Tokushima University and specially to **Dr. Kazuyuki Matsumoto**, **Dr. Mohamed abd Elfatah**, **Dr. Elsayed Atlam**, **Dr. Mahmoud Badee** and **Dr. Sohrab** for educational utilities, help and friendship.

I am also grateful to my family (my beloved Wife **Doaa Sabar**, my beloved daughters **Marim Elmarhoumy**, **Ment Allah Elmarhoumy** and my son **Ibrahim Elmarhoumy**), I could not complete this work without the unwavering love, patience, prayer, supplication (Doa'a) of them. My deep thanks to all of them for their patience and understanding during many hours of working and preparing the Information retrieval studies and for many years of living as foreigners outside our home country Egypt.

I would like to make a deep thank to my Parents (my beloved Father **Ibrahim Elmarhoumy** who is retired now even though he helped and supported me financial a lot till the last time of my Master thesis and I ask ALLAH to bless him in his life and put all my good deed in his account and enter him **Paradise** and my beloved Mother **Fadya Elmarhmoumy** for her supported me at all by what I need from her and by her doa'a.

Also, my deep thanks to my father in law **Elhaj Sabar**, my loved sister **Ghada Elmarhmoumy** and her children (**Moemen, Yomna, Mahmoud and Shahd Atlam**), she is always supporting me by her doa'a and advice, and also my brother **Ahmed Elmarhoumy** and his wife, and my Sister **Marwa Elmarhoumy** and her husband (**Dr. Salah ELzohary**).

My thanks is come to Mr. **Outera** and Mr. **Matshida** for their continuous support throughout the whole period of my life in Japan.

Also, all my greeting is to all Muslim friends in Japan and especially in **Tokushima** for their concern and encouragement.

I would like to thank all my Brothers In Islam in my country and my home town **Qohafa** for their help, brotherhood and doa'a to me at all.

I am thankful to all who knows me from near or far and had wished me the success.  
Finally, I would like to thank the staff of our Embassy in Japan for their encouragement  
and concern during my staying in Japan.

# **Bibliographic Notes**

Portions of this thesis are joint work and have appeared elsewhere based on the following papers.

## **Journal Paper**

Elmarhoumy M., Abdel Fattah M., Suzuki M., Ren F., “A New Modified Centroid Classifier Approach for Automatic Text Classification”, IEEJ TRANSACTIONS ON ELECTRICAL AND ELECTRONIN ENGINEERING, Volume 8, Issue 4, pp. 364-370, 2013, published.

Elmarhoumy M., Ren F., “A New Hybrid Model for Automatic Text Classification”, in The Online Journal on Computer Science and Information Technology (OJCSIT), Vol. (3)– No. (2), pp. 132-137, 2012.

## **International Conferences Papers**

Elmarhoumy M., Ren F., “A New Hybrid Model for Automatic Text Classification”, in World Congress on Computer Science and Information Technology, WCSIT’12, December 23-27, 2012.

Elmarhumy M., Abdel Fattah M., Ren F., “Automatic Text Classification Using Modified Centroid Classifier”, International Conference of IEEE-NLP-KE 2009, in the Proceedings of International Conference on Natural Language Processing and Knowledge Engineering, pp. 282-285, Dalian, china, Sep.24-27, 2009.

# CONTENTS

<b>Acknowledgements</b> .....	III
<b>Bibliographic Notes</b> .....	VI
<b>Contents</b> .....	VII
<b>List of Figures</b> .....	X
<b>List of Tables</b> .....	XI
<b>Abstract</b> .....	XII

## CHAPTERS

<b>1. Introduction</b> .....	1
<b>1.1 Automatic Text Classification</b> .....	2
<b>1.2 Motivation</b> .....	2
<b>1.3 Problem Description</b> .....	4
<b>1.4 Contribution</b> .....	5
<b>1.5 Prerequisite Concepts</b> .....	6
<b>1.5.1 Machine Learning Techniques</b> .....	6
<b>1.5.2 Text preparation</b> .....	8
<b>1.5.2.1 Stop-Word Elimination</b> .....	9
<b>1.5.2.2 Stemming</b> .....	9

1.5.3 Evaluation .....	10
1.5.3.1 Multiple Binary Classification Tasks .....	10
1.5.3.2 Multi-class and Multi-label Classification .....	12
1.6 Outline .....	13
2. Traditional Classifiers .....	15
2.1 Centroid Classifier Model .....	15
2.1.1 Similarity Measurement .....	17
2.1.2 Centroid Classifier System Outline .....	17
2.1.3 Traditional Centroid Classifier Problem .....	19
2.2 N-gram Word Based- Model .....	21
2.3 Naive Bayes .....	22
2.4 K-nearest Neighbour .....	23
2.5 Support Vector Machines .....	24
2.5.1 Training the SVM: the separable case .....	25
2.5.2 Training the SVM: the non-separable case .....	26
2.5.3 Ability to Incorporate New Documents .....	27
2.6 Term Weighting Method .....	27
3. Proposed Automatic Classifiers Techniques .....	29
3.1 Modified Centroid Classifier .....	29
3.1.1 Modified centroid classifier system outline .....	31
3.2 A New Term Weight .....	33
3.3 Enhancing N-gram Word Based Model .....	39
3.4 The Hybrid Model Based Center Profile Vector .....	40



3.4.1 The CPV Model Training Step .....	41
3.4.2 The CPV Model Testing Step .....	41
4. Evaluation and Experimental Results .....	43
4.1 Modified Centroid Classifier .....	43
4.1.1 Centroid Classifier Accuracy .....	44
4.1.2 Modified Centroid Classifier Accuracy .....	44
4.2 Accuracy Comparison for Proposed and Traditional Approaches .....	46
4.3 The Hybrid Model and Modified N-gram Model .....	47
5. Conclusion & Future Work .....	57
5.1 Conclusion .....	57
5.2 Future Work .....	59
References .....	61

# List of Figures

<b>1.1 Trained Model.....</b>	<b>9</b>
<b>2.1 Outline of Traditional Centriod Classifier System .....</b>	<b>17</b>
<b>2.2 A two Class Text Data Distribution .....</b>	<b>20</b>
<b>2.3 N-gram Classes and Documents Representation .....</b>	<b>21</b>
<b>3.1 Modified Centroid Classifier .....</b>	<b>31</b>
<b>3.2 Outlines of the Modified Centriod Classifier System .....</b>	<b>32</b>
<b>4.1 Changing accuracy with threshold values .....</b>	<b>45</b>
<b>4.2 Comparison among Different Approaches based on Reuter-21578 Corpus ...</b>	<b>47</b>
<b>4.3 the F-measure for each class using traditional NB, SVM, N-gram, DN-g, TC, AC and NC classifiers .....</b>	<b>49</b>
<b>4.4 F-measure for Each Class Using SN-g, MN-g, CPV&amp;T, CPV&amp;A and CPV&amp;N Classifiers based on the term weighting <math>\frac{TF_{i,k}}{TF_{i,c}^*}</math> .....</b>	<b>52</b>
<b>4.5 F-measure for Each Class Using SN-g, MN-g, CPV&amp;T, CPV&amp;A and CPV&amp;N Classifiers based on the term weighting <math>\frac{DF_{i,k}}{DF_{i,c}^*}</math> .....</b>	<b>53</b>
<b>4.6 F-measure for Each Class Using SN-g, MN-g, CPV&amp;T, CPV&amp;A and CPV&amp;N Classifiers based on the term weighting <math>\frac{TS_{i,k}}{TS_{i,c}^*}</math> .....</b>	<b>53</b>
<b>4.7 F-measure for Each Class Using SN-g, MN-g, CPV&amp;T, CPV&amp;A and CPV&amp;N Classifiers based on the term weighting <math>\frac{DS_{i,k}}{DS_{i,c}^*}</math> .....</b>	<b>54</b>
<b>4.8 F-measure for Each Class Using SN-g, MN-g, CPV&amp;T, CPV&amp;A and CPV&amp;N Classifiers based on the term weighting <math>tfsc, dfsc</math> .....</b>	<b>55</b>

# List of Tables

Table 1. The training and testing data.....	43
Table 2. Document Classification Accuracy using Traditional Centroid Classifier ..	44
Table 3. Document Classification Accuracy using Modified Centroid Classifier ...	45
Table 4. Statistical Significance for Total Performance of Traditional Centroid Classifier and Modified Centroid Classifier .....	46
Table 5: F-measure for Each Class Using Traditional NB, SVM, N-gram, DN-g, TC, AC and NC Classifiers .....	48
Table 6: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting $\frac{TF_{i,k}}{TF_{i,c}^*}$ .....	50
Table 7: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting $\frac{DF_{i,k}}{DF_{i,c}^*}$ .....	51
Table 8: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting $\frac{TS_{i,k}}{TS_{i,c}^*}$ .....	51
Table 9: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting $\frac{DS_{i,k}}{DS_{i,c}^*}$ .....	52
Table 10: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting $tfsc, dfsc$ .....	55

# ABSTRACT

To enhance the automatic text classification task, we propose a novel approach for treating the problem of inductive bias incurred by the centroid classifier assumption. This approach is a trainable classifier, which takes into account *tfidf* as a text feature. The main goal of the proposed approach is to take advantage of the most similar training errors in the classification model for successively updating that model based on a certain threshold. The proposed approach is practical and flexible to implement. The complete performance of the proposed approach is measured at several threshold values on the Reuters-21578 text categorization collection. Experimental results show that the proposed approach can improve the performance of the centroid classifier better than traditional approaches (traditional centroid classifier, support vector machines, decision trees, Bayes nets, and N Bayes) by 1, 1.2, 4.1, 7.5, and 11%, respectively.

Moreover, we present a new hybrid Center Profile Vector (CPV) classification model based on the modified N-gram and centroid classifier models. The hybrid model exploits a new approach to calculate the term weight based on *tfsc, dfsc* in order to sort the terms in the model profile. Moreover, we present a new distance similarity method for the N-gram model that solves the problem of the difference in representation lengths among classes and documents. The hybrid (CPV) classification model provides promising classification rate compared with some other models such as N-gram Model, Centroid Classifier, Naive Bays and Support Vector Machine.

# CHAPTER 1

## INTRODUCTION

Automatic Text Classification is very important task for the management information applications by assignment of automatic texts to one or more predefined categories, it could be used for a lot of management information applications such as an indexing mechanism for text retrieval and a component of an information filtering system. By 90s, the machine learning technique has been used to build the automatic text classifier by using the training set of document, this type of classifier has gained a high popularity due to its advantages. To build the classifier model, traditional methods used set of pre-classified document from training set of document to train the classifier model to be ready to classify any document. In the last few years the machine learning techniques has led to enhance the performance of classifier model by saving the time and straightforward portability to different domains.

Considerable research has gone into text classification, using many approaches, such as the nearest neighbor method [8, 9] and the Rocchio classifiers model [10, 11] which classifies full-text news stories and reaches high recall with moderated precision, without requiring manual definitions of the various topics. Other algorithms, such as decision tree models [12], focus on automatic text categorization and developing predictive capabilities, e.g.

The size of nowadays text collections usually exceeds the size of the datasets that current software handles properly [35-37]. Therefore, data pre-processing is one of the most critical steps in text classification techniques, performed to ensure good quality data for classification.

Feature selection has been studied in the text categorization literature [38, 39]. Feature selection is used to reduce dimensionality.

## **1.1 Automatic Text Classification**

Text classification is the process of automatically classify texts that provides useful information for the user. Document classification/categorization is a problem that assigns an electronic document to one or more class depends on its contents. Document classification tasks are divided into two groups: the first group is called supervised document classification that use human feedback to provide information on the correct classification for documents; the other group is called unsupervised document classification, where the classification must be done entirely without reference to human feedback. Moreover, there is also a semi-supervised document classification, where parts of the documents are labeled by the external mechanism.

## **1.2 Motivation**

Nowadays, many documents are processed automatically by a computer. With the growth of the Internet, the number of the online documents has become huge [1]. Therefore, development of automated text classification systems has become a serious and important

issue in organizing these documents [2]. A number of techniques for text classification have been viewed as supervised learning, ranging from those situations where the target has predefined class labels to unlabeled documents where organization is based on similarity deduction from the training set of labeled documents [3-7]. Considerable research has gone into text classification, using many approaches, such as the nearest neighbor method [8,9] and the Rocchio classifiers model [10, 11] which classifies full-text news stories and reaches high recall with moderated precision, without requiring manual definitions of the various topics. Other algorithms, such as decision tree models [12], focus on automatic text categorization and developing predictive capabilities, e.g. Bayesian probabilistic algorithms [13, 14]. This approach examines inductive learning to classify natural language texts into predecessor content categories and the support vector machines (SVMs) method [15, 16]. It consists of a set of related supervised learning methods that analyze data and arrange patterns for classification. However, all these approaches involve large computations in both learning and testing processing; this is very expensive, as most of these algorithms use complicated processes.

## **1.3 Problem Description**

A centroid-based approach is a linear model in text classification that is a more practical and efficient approach than previous approaches because it involves smaller computations than the earlier approaches in terms of both learning and testing processing [17]. Many centroid-based methods [18, 19] have been used in text classification applications [20-22]. However, these methods often suffer from inductive bias or model misfit [23] incurred by their assumption. This assumption is often violated when existing

documents from a certain class share more similarity with the centroid of other classes than of its own class [34]. The most difficult problem regarding model misfit is the poor classification performance [9, 44 and 32]. A weight adjustment technique for the centroid classifier was proposed by Shankar and Karypis [46]. This approach gradually adjusts the weights of all features concurrently, using the discriminating power of each term. This technique assumes that terms with higher discrimination will be more important in classification than terms with lower discrimination. The choice of the best set of parameters for the centroid model can be found using empirical experimentation; an approach to avoid misfits is required. Some researchers have investigated the development of methods (such as Drag Pushing [32]) to improve the centroid classifier.

Moreover, there are some drawbacks of the classic *tfidf* term weighting scheme for the text classification task. In the training vector space, to compute the weight of a certain term, the category information is constantly omitted by the document-indexing-based *tfidf* term weighting method. In contrast, the inverse document frequency (*idf*) function provides the lowest score of those terms that appear in multiple documents; because of this, the *tfidf* score gives positive discrimination to rare terms and is biased against frequent terms. At present, because *tfidf* uses a default term weighting parameter in the classification task, a variety of feature selection techniques, such as information gain [45], chi-square test, and document frequency [39], have been used to reduce the dimension of the vectors. Therefore, a novel term weighting method is needed to overcome the problem of high dimensionality and for the effective indexing based on class to enhance the classification task.



## 1.4 Contribution

In order to solve this problem, we propose an approach that takes advantage of the most similar training errors in the classification model to successively update it at a certain threshold. The proposed approach moves the centroid of each class by a certain distance, based on the misclassified documents of each class that have maximum similarities with the particular class. Considering only training errors that have maximum similarities with the particular class increases the accuracy of classification. The experimental results using Reuters-21578 text categorization collection indicate that the proposed approach can improve the performance of the centroid classifier more than traditional approaches (traditional centroid classifier, SVMs, decision trees, Bayes nets and N Bayes) by 1, 1.2, 4.1, 7.5, and 11%, respectively [47].

Many of the previously mentioned approaches are based on traditional models and *tfidf* to weigh a term in the document. In this work, a new term weight called “*tfsc,dfsc*” is developed. This term weight is based on term frequency in the class, document frequency in the class, term distribution in the class, document distribution in the class, term frequency in other classes, document frequency in other classes, term distribution in other classes and document distribution in other classes. A combination of “*tfidf*”, “*tfsc,dfsc*” besides the term it self are exploited to create the hybrid model based on the modified N-gram and centroid classifier models. The results of the proposed approach are promising.

A new distance similarity measure is applied for the modified N-gram model to calculate the distance similarity. This new distance similarity measure solves the problem of difference of the profile lengths among the classes and between the document and the

class. The new distance formula helps the model to classify the documents accurately. Empirically, the longest class profile contains 4451 words, and the shortest class profile contains 2714 words. That means there is high difference among classes. In this case, the Manhattan distance will be  $(4451-2714=1737)$ , which has bad effect on the classification task. However, when we use the new distance formula, the highest length difference will be  $(\log 4451 - \log 2714 = 0.215)$ . Therefore, the new distance measure will have a good effect on the performance of the modified N-gram. Taking the modulus of  $\log (c/d)$  makes the distance value ranges from 0 to 3.7, which solves the problem of the distinction between the query document profile length and the class profile length [48,49].

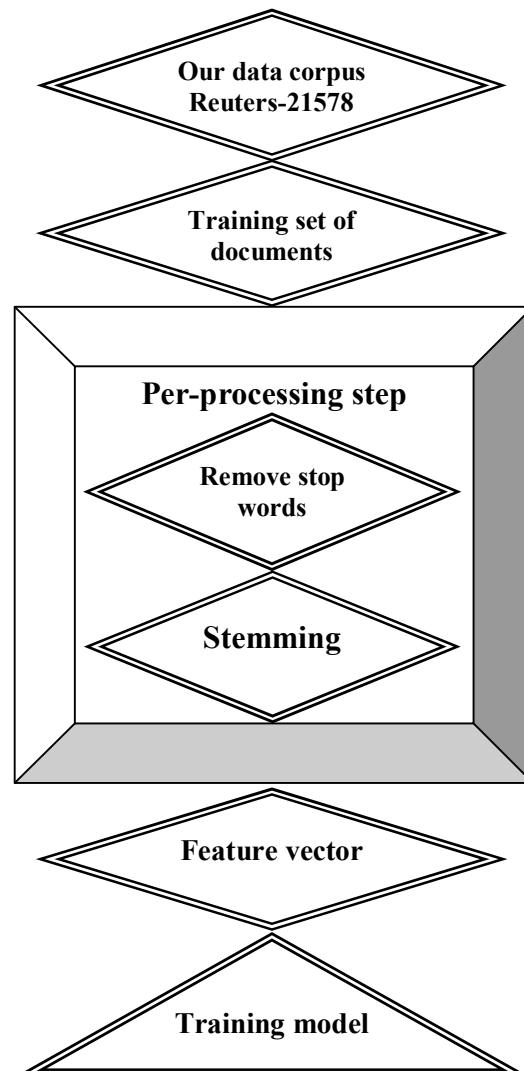
Moreover, we proposed a novel hybrid CVP classification model. The proposed CPV model will gain the benefits of the centroid model and the modified N-gram model to achieve high classification rate as will be shown in the experimental result. Moreover, the proposed approach uses two weighting approaches at the same time, which makes the proposed PCV model able to gain the benefits of two weighting approaches simultaneously.

## **1.5 Prerequisite Concepts**

### **1.5.1 Machine Learning Techniques**

Machine learning is a broad subfield of artificial intelligence, which is concerned with the design and development of algorithms and techniques that allows computer to learn. The major focus of machine learning research is to extract information from data

automatically, by computational and statistical methods (Croft and Harper [50], Fukumoto, [51]; Vapnik, [52]; Yang and Pedersen, [53]; Yang, [54]; Friedman [55]; Yang, [56]; [13]).



**Fig. 1.1 Trained Model**

This technique is based on a set of document and their predefined categories. The process is then modeled as a classification problem where documents are categorized as a set of predefined categories, based on feature selection method. The classification probabilities arise statistically from the training data, using Baye's [4] formula:

$$P(C_1, \dots, C_n | F_1, \dots, F_n) = \frac{P(F_1, \dots, F_n | C_1, \dots, C_n) P(C_1, \dots, C_n)}{P(F_1, \dots, F_n)} \quad (1.1)$$

Where,  $\{C_1, \dots, C_n\}$  means  $n$  classes, as well as  $\{F_1, \dots, F_n\}$  means  $n$  feature sequence.

The probability  $P(F_1, \dots, F_n | C_1, \dots, C_n)$  can be estimated as  $\prod_{i=1}^n P(F_i | C_i)$

Fig. 1.1 trained our model by using Reuters-21578 collection. In Fig. 1.1, the training set of documents is used as input data. Second making per-processing for these training documents by: removing stop words that will explain in section 1.5.2.1 and stemmed words that will describe in section 1.5.2.2. Third the feature vector for the training documents prepared. Finally our model is trained by feature vector for the training documents to construct a prototype vector, or centroid, per class which will investigate in details in chapter 2.

## 1.5.2 Text preparation

The first step in text categorization is to transform documents, which typically are strings of characters, into a representation suitable for the learning algorithm and the classification task. The text transformation usually is as the following three kinds:

- Remove HTML (or other) tags
- Remove stop words
- Perform word stemming

### 1.5.2.1 Stop-Word Elimination

In this subsection we will explain in detail how we can eliminate the un-useful words. Among many words, some words are too frequent to work as a useful feature. For example, the verbs “be” and “has” can be seen in almost any documents. Such words are called *stop-words* and often removed from the feature set. One problem in *stop-word* elimination is that a word can be a stop-word for a data set, but can be a useful feature for another data set.

The stop words are frequent words that have little meaning and information (i.e. pronouns, prepositions, conjunctions etc).

### 1.5.2.2 Stemming

By word stemming we mean the process of suffix removal to generate word stems. This is done to group words that have the same conceptual meaning, such as walk, walker, Walked, and walking. The Porter stemmer [57, 58 and 60] is a well-known algorithm for this task. Stemming is the process for reducing inflected words of their stem, base or root form, generally a written word form. Stemming is widely using in search engine [59] technology for query expansion [61] and Natural Language Processing [62, 63 and 43]. For text classification, in our experiment we used Porter stemming algorithm.

This stemming process make the data corpus more effective to further use for identify word frequency count and tagging documents either it's overlap or non overlap word. The most beneficial factor for using stemming algorithm is that, as stemming algorithm can addresses base or root form so it can keep avoid to count a certain root word from multiple times. So it may increase the data corpus accuracy.

### **1.5.3 Evaluation**

An important issue of text categorization is how to measure the performance of the classifiers. Many measures have been used, each of which has been designed to evaluate some aspect of the categorization performance of a system. In this chapter we describe some of the measures that have been reported in the literature, section 1.5.3.1 treats the multiple binary classification tasks, while section 1.5.3.2 describes measures that have been used for evaluating the performance of the multi-class methods.

#### **1.5.3.1 Multiple Binary Classification Tasks**

A common approach for multi-class categorization [12] is to break the task into disjoint binary categorization problems. For each category and each document one determines whether the document belongs to the category or not. When evaluating the performance of the classifiers, four quantities are of interest for each category:

- $a$  - the number of documents correctly assigned to this category,
- $b$  - the number of documents incorrectly assigned to this category,

- $c$  - the number of documents incorrectly rejected from this category,
- $d$  - the number of documents correctly rejected from this category.

From these quantities, we define the following performance measures called Recall and Precision [57]:

$$\text{Recall} = \frac{a}{a + c} \quad (1.2)$$

$$\text{Precision} = \frac{a}{a + b} \quad (1.3)$$

$$\text{Fallout} = \frac{b}{b + d} \quad (1.4)$$

$$\text{Accuracy} = \frac{a + d}{a + b + c + d} \quad (1.5)$$

$$\text{Error} = \frac{b + c}{a + b + c + d} \quad (1.6)$$

**Micro- and macro-averaging:** For evaluating performance average across categories, there are two conventional methods, namely *macro-averaging* and *micro-averaging* [64]. Macro- averaged performance scores are determined by first computing the performance measures per category and then averaging these to compute the global means. Micro-average performance scores are determined by first computing the totals of  $a, b, c$  and  $d$  for all categories and then use these totals to compute the performance measures. There is an important distinction between the two types of averaging. Micro-averaging gives equal weight to every document, while macro-averaging gives equal weight to each category.

F-measure Another evaluation criterion that combines recall and precision is the F-measure:

$$F\beta = \frac{(\beta^2 + 1) \times \mathbf{precision} \times \mathbf{recall}}{\beta^2 \times \mathbf{precision} + \mathbf{recall}} \quad (1.7)$$

where  $\beta$  is a parameter allowing different weighting of recall and precision.

Interpolation For some methods, the category assignments are made by threshold a confidence value. For instance the Bayes classifier computes the probability that a document is in the current category, and one has to decide how large this probability must be (specified by a threshold) for the document to be assigned to the category. By adjusting this threshold, one can achieve different levels of recall and precision. In the results for different thresholds are combined using interpolations.

### 1.5.3.2 Multi-class and Multi-label Classification

To measure the performance of a classifier that produces a ranked list of categories for each test document, with a confidence score for each candidate, an approach called *interpolated. 11-point average precision* may be used. In this approach the recall for one specific document is defined to be:

$$\text{Recall} = \frac{\text{Number of categories found that are correct}}{\text{Total number of true categories}} \quad (1.8)$$



For each of 11 values 0.0, ..., 1.0 of this fraction, the system decides how far down the ranked list one has to go ( i.e. the size of the numerator ) to achieve the specified recall.

The precision is then computed for this number of categories by:

$$\text{Precision} = \frac{\text{Number of categories found that are correct}}{\text{Total number of categories found}} \quad (1.9)$$

The resulting 11 precision values are averaged to obtain a single number-measure of performance for the document. For a test set of documents, the average precision values of individual documents are further averaged to obtain a global measure of system performance over the entire set.

## 1.6 Outline

This thesis presents research on Automatic Classifiers Techniques based on text classification a new approaches is an important addition to the field of classification to meet an increasing of online information demand. The thesis is divided into 5 chapters: chapter 1 gives the introduction, of text classification system and explains the automatic text classification, motivation, problem description, contribution, prerequisite concepts, classification evaluation and outline of the thesis. Chapter 2 gives an overview of text pre-process algorithms to build for training and testing data and describes different methods and algorithms for trainable classifier. Chapter 3 deals with a concrete idea and proposed methods, algorithms and models for automatic text classification using Centroid Classifier and n-gram. Chapter 4 looks the experimental results on different text features

and the result to compare with the baseline method to the proposed methods. Finally, Chapter 5 focuses on conclusion and future work.

# CHAPTER 2

## TRADITIONAL CLASSIFIERS

This chapter describes the traditional classification models which are centroid classifier model and N-gram model based on word. Moreover, the end of this section shows the applying of weighting method.

### 2.1 Centroid Classifier Model

Let's take a set of classes  $K = \{k_1, k_2, \dots, k_m\}$  and the training data set of documents  $D = \{d_1, d_2, \dots, d_S\}$  where the training document  $d_i$  should be assigned to one class, by using this given information, classifier can find one suitable class for a new document. Vector space model represents documents and classes as vectors based on the tfidf weight of each term in the document or class. In centroid classifier, documents and classes are represented by using vector space model (VSM) which considers each document and class as a vector in the term-space. We may exploit centroid in three forms. The first one is to create the centroid of the class by summing all the term values in the document vectors that are related to one class as in formula 1. This form is called traditional centroid. The second one is to create the centroid of the class by taking the average of summing all term values in the document vectors of the class divided by number of the documents in the class as in formula 2. This form is called average centroid. The third one is to create the centroid of the class by taking the normalized value of summing all

term values in the document vectors of the class as in formula 3, This form is called normalized centroid. The three types of centroid classifier are established by using the following equations:

### **Traditional Centroid Classifier**

$$C_k = \sum_{d \in C_k} d \quad (2.1)$$

Where  $C_k$  is the traditional centroid vector of class  $C_k$  and  $\sum_{d \in C_k} d$  is summation of all training document vectors  $d$  in class  $C_k$ .

### **Average Centroid Classifier**

$$AC_k = \frac{1}{dn_k} \left( \sum_{d \in C_k} d \right) \quad (2.2)$$

Where  $AC_k$  is the Average centroid vector of class  $C_k$ ,  $dn_k$  is number of documents in class  $C_k$ .

### **Normalized Centroid Classifier**

$$NC_k = \frac{C_k}{\|C_k\|_2} \quad (2.3)$$

Where  $NC_k$  is the normalized centroid vector of class  $C_k$  and  $\|C_k\|_2$  denotes the 2-norm of vector  $C_k$  [13].

### 2.1.1 Similarity Measurement

After creating a centroid for each class, the query documents can be classified based on the similarity measure. In centroid classifier similarity can be calculated by applying dot product between query document vector and class centroid vector. Therefore, the query document is classified as a certain class if the centroid vector of this class is the most similar to the query document vector according to the following formula:

$$c_m = \underset{c_n \in C}{\operatorname{argmax}} \operatorname{sim}(d_t, c_n) = \underset{c_n \in C}{\operatorname{argmax}} d_t \cdot c_n \quad (2.4)$$

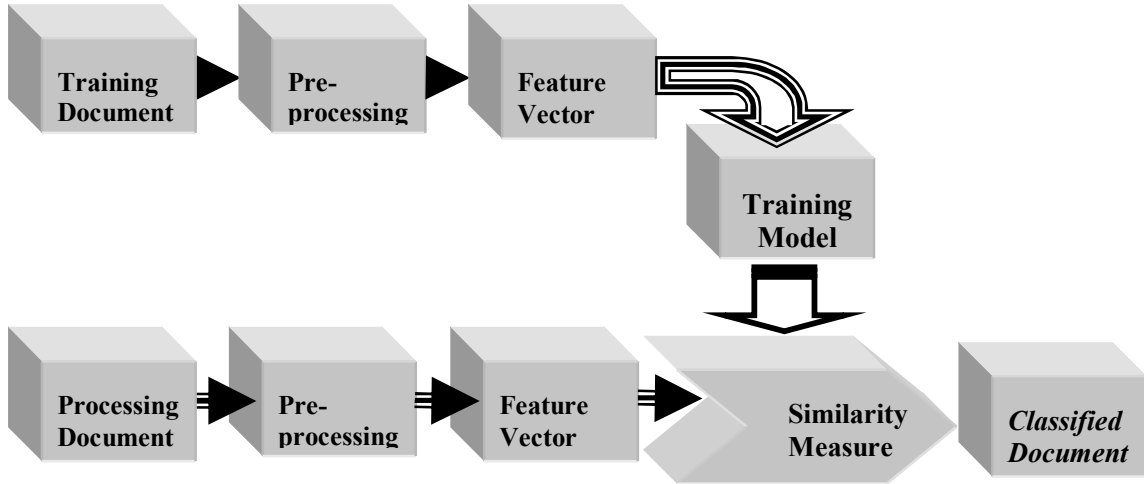


Fig. 2.1 Outlines of Traditional Centroid Classifier System

### 2.1.2 Centroid Classifier System Outline

Figure 2.1 shows the outline of the traditional approach. From Fig. 2.1, it is clear that the following steps were investigated with two processing, A and B:

## **Processing A: Training Centroid Classifier**

### ***Training Documents:***

Prepare the training documents from the corpus.

### ***Preprocessing:***

The Stop Word List 1' at (<http://www.lextek.com/manuals/onix/stopwords1.html>) has been used as the stop words. Light stemmer has been used. After removal of the stop word, all stemmed words have been exploited to create *tfidf* feature parameter vectors in the experiments.

### ***Feature Vector:***

Create the document vectors using *tfidf*.

### ***Training Model:***

The formatted input file, as well as the prototype vector or centroid for each class, was created based on (2.3); model is ready to classify any document, as shown in Fig. 2.1.

## **Processing B: Testing Centroid Classifier**

### ***Testing Documents:***

Prepare the testing documents from the corpus.

### ***Preprocessing:***

Remove stop words and extract the stemmed word as in the preprocessing of training step.

### ***Feature Vector:***

Create the document feature vectors using *tfidf* as in the training step.

### ***Similarity Measure:***

In this step, the similarity between the testing documents vectors and the centroid vector for each class is evaluated by using the inner product between the class vector and the query vector.

### ***Classified Document:***

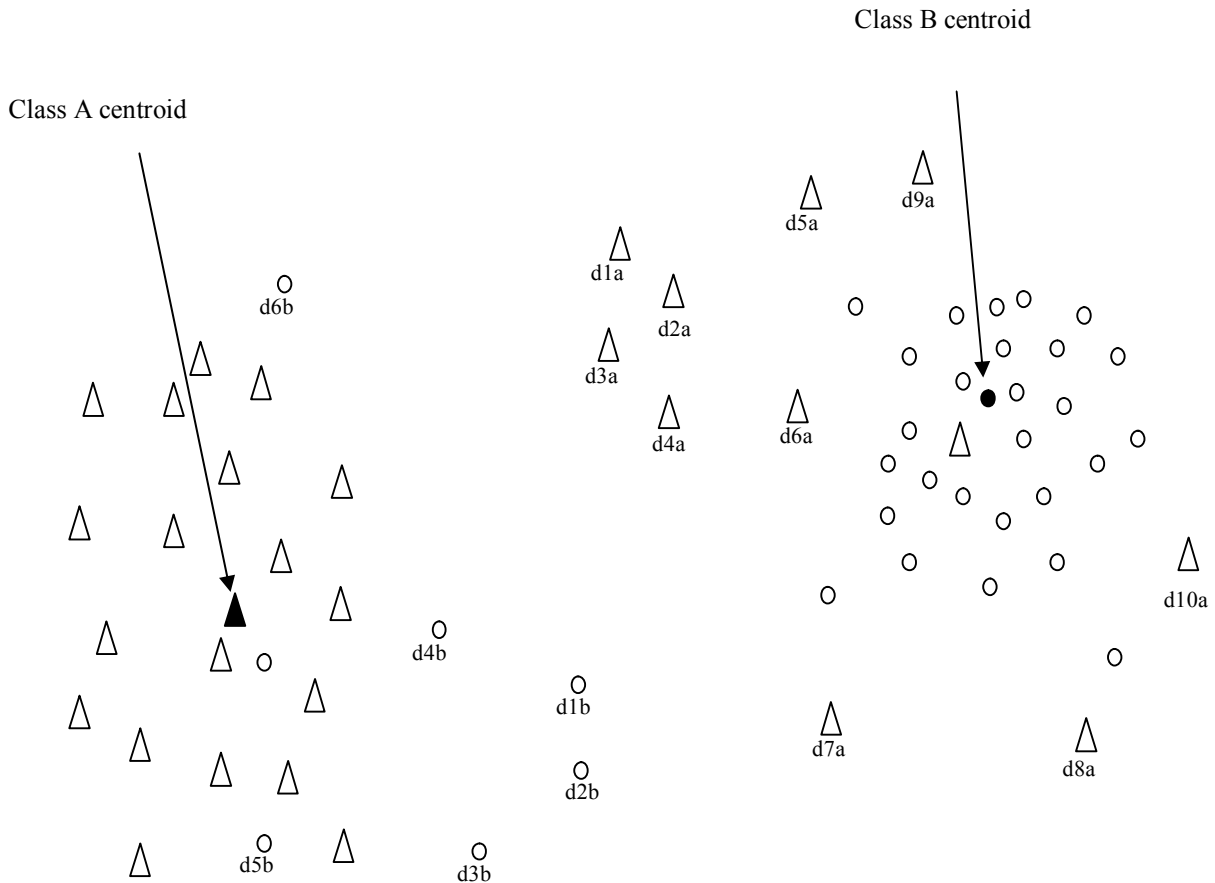
The system-classified documents assigned to the class  $c_i$  whose class prototype vector is the similar to that of the document described by formula (2.4).

## **2.1.3 Traditional Centroid Classifier Problem**

In centroid classifier, it is supposed that a given document should be assigned to a particular class if the similarity between this document and that centroid is the largest. Nevertheless, this supposition is often violated when documents from a certain class sharing more similarities with the centroids of other classes.

Fig. 2.2 shows two classes of text data: Class A spreads in a triangle shape; Class B spreads in a circle shape. Obviously, the examples (d1a to d3a) are correctly classified as category A since they share more similarities with centroid A rather than with centroid B. However, d4a to d10a will be misclassified into class B since they share more similarities with centroid B than with centroid A. Adding the training errors (d4a to d10a) to centroid A to adjust its prototype vector will move centroid A toward centroid B by a large distance. This process will move centroids A and B closer to each other, which will definitely result in deterioration of the total system performance. Moreover, adding the

training errors (d4a to d10a) to centroid A and subtracting them from centroid B will result in deterioration of the total system performance as well. Unfortunately, as shown in the examples of Fig. 2.2, centroid A will move toward the original documents of class B, while centroid B will move away from its original document; these results in some of the class B original documents being misclassified into class A. The same problem occurs for class B documents that share more similarity with centroid A than with centroid B. The proposed method will solve this problem using a modified centroid.

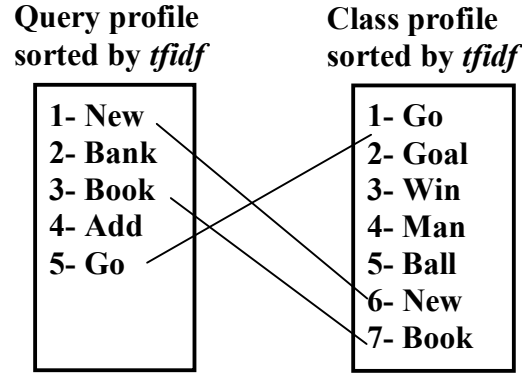


**Fig.2.2 A Two Class Text Data Distribution**



## 2.2 N-gram Word Based- Model

In N-gram word based- model, the class  $c_k$  is represented as a class profile  $k$ . The class profile  $k$  contains all the words inside all the training documents which are related to class  $C_k$ . The words inside the class profile are sorted according to the highest *tfidf* score. Likewise, the query document  $d$  is represented as a query profile  $d$ . This query profile  $d$  presents all the words in the query document sorted based on the highest *tfidf* score [7]



**Fig. 2.3 N-gram Classes and Documents Representation**

In N-gram model the similarity will be computed by measuring the distance between the representations of query document profile  $d$  and each class profile  $k$ . For each word in a query document profile, we find its counterpart in the class profile and compute the number of places its location differs. Based on Fig. 2.3 the distance similarity between the query document and the class will be as follows: in query profile the word [New] is ranked as 1, the same word in the class profile is ranked as 6, the distance will be  $|1-6| = 5$ , but for the word [Bank] in the query profile, it has no counterpart in the class profile. Therefore, the distance is 7, the maximum distance {the

total number of the terms inside the class profile}. Similarly, the similarity distance of the word [Book] =  $|3-7| = 4$ , the distance similarity of the word [Add] = 7 and the similarity distance of the word [Go] =  $|5-1| = 4$ . The total distance  $(d,c) = 5+7+4+7+4 = 27$ . The query document is classified as the class profile which has minimum difference with the query profile, according to (Manhattan distance) as in the following formula:

$$c_i = \arg \min_{c \in C} dis(d,c) = \arg \min_{c \in C} (|d - c|) \quad (2.5)$$

## 2.3 Naive Bayes

The naive Bayes classifier is constructed by using the training data to estimate the probability of each class given the document feature values of a new instance. We use Bayes theorem to estimate the probabilities as follows:

$$P(c_j | d) = \frac{P(c_j)P(d | c_j)}{P(d)} \quad (2.6)$$

The denominator in the above equation does not differ between categories and can be left Out. Moreover, the naive part of such a model is the assumption of word independence, i.e. we assume that the features are conditionally independent, given the class variable. This simplifies the computations yielding:

$$P(c_j | d) = P(c_j) \prod_{i=1}^M P(d_i | c_j) \quad (2.7)$$

An estimate  $\hat{P}(c_j)$  for  $P(c_j)$  can be calculated from the fraction of training documents that is assigned to class  $c_j$  as:

$$\hat{P}(C = c_j) = \frac{N_j}{N} \quad (2.8)$$

Moreover, an estimate  $\hat{P}(d_i|c_j)$  for  $P(d_i|c_j)$  is given by:

$$\hat{P}(d_i|c_j) = \frac{1 + N_{ij}}{M + \sum_{k=1}^M N_{kj}} \quad (2.9)$$

where  $N_{ij}$  is the number of times word  $i$  occurred within documents from class  $c_j$  in the training set.

Despite the fact that the assumption of conditional independence is generally not true for word appearance in documents, the Naive Bayes classifier is surprisingly effective.

## 2.4 K-nearest Neighbour

To classify an unknown document vector  $d$ , the k-nearest neighbour (kNN) algorithm ranks the documents neighbours among the training document vectors, and use the class labels of the k most similar neighbours to predict the class of the input document. The classes of these neighbours are weighted using the similarity of each neighbour to  $d$ , where similarity may be measured by Euclidean distance or the cosine between the two document vectors.

kNN is a lazy learning instance-based method that does not have a offline training phase. The main computation is the on-line scoring of training documents given a test document in order to find the  $k$  nearest neighbours. Using an inverted-file indexing of

training documents, the time complexity is  $O(L \times N / M)$  where  $L$  is the number of elements of the document vector that are greater than zero,  $M$  is the length of the document vector, and  $N$  is the number of training samples.

## 2.5 Support Vector Machines

Support Vector Machines (SVMs) have shown to yield good generalization performance on a wide variety of classification problems, most recently text categorization. The SVM integrates dimension reduction and classification. It is only applicable for binary classification tasks, meaning that, using this method text categorization have to be treated as a series of dichotomous classification problems.

The SVM classifies a vector  $d$  to either -1 or 1 using

$$s = W^T \phi(d) + b = \sum_{i=1}^N \alpha_i y_i K(d, d_i) + b \quad (2.10)$$

And

$$y = \begin{cases} 1 & \text{IF } s > s_0 \\ -1 & \text{otherwise} \end{cases} \quad (2.11)$$

Here  $\{d_i\}_{i=1}^N$  is the set of training vectors as before and  $\{y_i\}_{i=1}^N$  are the corresponding classes ( $y_i \in -1, 1$ ).  $K(d_i, d_j)$  that denoted a kernel and is often chosen as a polynom of degree  $\bar{d}$ , i.e.

$$K(d, d_i) = (d^T d_i + 1)^{\bar{d}} \quad (2.12)$$

The training of the *SVM* consists of determining the  $W$  that maximizes the distance between the training samples from the two classes. In the following sub-sections we will describe these achievements.

### 2.5.1 Training the SVM: the separable case

First, consider the case for which the data is linearly separable. That means that there exists a vector  $W$  and a scalar  $b$  such that:

$$W^T \phi(d_i) + b \geq 1 \quad \text{if } y_i = 1 \quad (2.13)$$

$$W^T \phi(d_i) + b \leq -1 \quad \text{if } y_i = -1 \quad (2.14)$$

for all  $\{d_i\}_{i=1}^N$ . The SVM constructs a hyper plane  $W^T \phi(d) + b$  for which the separation between the two classes is maximized. The  $W$  for the optimal hyper-plane can be found by minimizing

$$\|W\|^2 \quad (2.15)$$

The optimal  $W$  can be written as a linear combination of  $\phi(d)$ 's, i.e.,

$$\sum_{i=1}^N \alpha_i y_i \phi(d_i) \quad (2.16)$$

where  $\{\alpha_i\}_{i=1}^N$  can be found by maximizing:

$$\Lambda^T 1 - \frac{1}{2} \Lambda^T Q \Lambda \quad (2.17)$$

with respect to  $\Lambda$ , under the constraints

$$\Lambda \geq 0 \quad \text{and} \quad \Lambda^T Y = 0 \quad (2.18)$$

Here  $Y = (y_1, \dots, y_N)$  and  $Q$  is a symmetric matrix with elements

$$Q_{ij} = y_i y_j K(d_i, d_j) = y_i y_j \phi(d_i)^T \phi(d_j) \quad (2.19)$$

Only the  $\alpha_i$ 's corresponding to the training examples along the margins of the decision boundary (i.e. the support vectors) are greater than zero.

### 2.5.2 Training the SVM: the non-separable case

In the non-separable case, equation (3.14) is altered to:

$$\frac{1}{2} \|W\|^2 + C \sum_{i=1}^N \xi_i \quad (2.20)$$

where  $\xi_i$  satisfy the constraints:

$$W^T \phi(d_i) + b \geq 1 - \xi_i \quad \text{if } y_i = 1 \quad (2.21)$$

$$W^T \phi(d_i) + b \leq -1 + \xi_i \quad \text{if } y_i = -1 \quad (2.22)$$

The user-defined parameter  $C$  balances the contributions from the first and second terms. Minimization of Equation (3.19) can be achieved by maximizing Equation (3.16) under the constraints

$$0 \leq \Lambda \leq C1 \quad \text{and} \quad \Lambda^T Y = 0 \quad (2.23)$$

### 2.5.3 Ability to Incorporate New Documents

The optimization problem described above is very challenging when the data set is large, as the memory requirement grows with the square of the size of the data set. The computational and storage complexities can be reduced by dividing the training set into a number of *chunks* and extracting support vectors from each set. The support vectors can later be combined together.

The same procedure can be used for incorporating new documents into the existing set of support vectors. It can be shown that the final solution is as good as if all documents were processed together.

## 2.6 Term Weighting Method

The most famous weighting algorithm *tfidf* is exploited in our experimentation to establish the term weighting. The *tfidf* is calculated according to the following formula:

$$tf_{r,m} = \frac{t_{r,m}}{\sum_n t_{n,m}} \quad (2.24)$$

Where  $t_{r,m}$  is time of occurrences of term  $t_r$  in document  $d_m$ , and the denominator is the sum of number of occurrences of all terms in document  $d_m$ .

$$idf_r = \log \frac{|D|}{|d_r|} \quad (2.25)$$

Where  $|D|$  is the total number of documents in the corpus and  $|d_r|$  is the number of documents that contain term  $t_r$ .

$$tfidf_{r,m} = tf_{r,m} \times idf_r \quad (2.26)$$



# CHAPTER 3

## PROPOSED AUTOMATIC CLASSIFIERS TECHNIQUES

### 3.1 Modified Centroid Classifier

This section describes the building process of the modified centroid classifier and the outlines of the modified model. Then we explain the proposed idea of the modified centroid classifier and introduce the proposed formula.

In the centroid classifier, it is supposed that a given document is assigned to a particular class if the similarity between the document and the centroid is the largest. Nevertheless, the supposition is often violated when assigning documents from a certain class that shares more similarities with the centroids of other classes.

To overcome the drawback of the traditional centroid classifier methods, we propose a modified centroid classifier model. The proposed model adds the most common training errors in a certain class to its centroid to update it discards the training errors that have few similarities with their class (by not including them while updating the centroid) based on a certain threshold value according to the following formula:

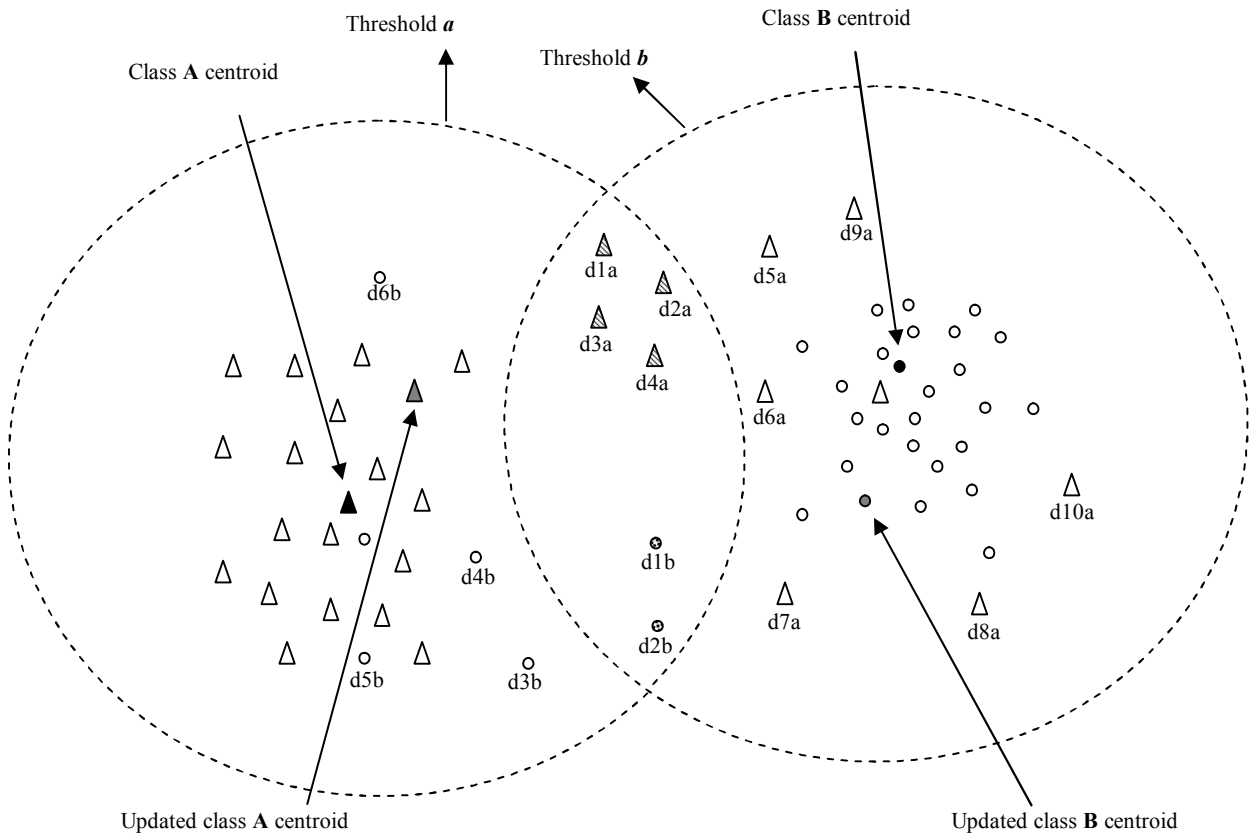
$$C_{i\_modified} = C_i + \frac{\sum_{d \in \text{class } i \text{ \& classified as other categories and has similarity with } C_i > \text{threshold}} d}{\sum_{d \in \text{class } i \text{ \& classified as other categories and has similarity with } C_i > \text{threshold}} \|d\|_2} \quad (3.1)$$

where  $C_i$  is the normalized centroid calculated from formula (2.3). Based on formula (3.1), by picking out the misclassified documents by categorizing all training sets and appending the most similar misclassified training errors to their correct classes based on a certain threshold value, the corresponding centroid is updated. In the modified centroid classifier model, the most similar training errors belonging to a certain class are added to its centroid for updating, and the training errors that have low similarities with their class based on a threshold value according to formula (3.1) are discarded. Using formula (3.1) and (2.4) after substituting  $C_i$  with  $C_{i\_modified}$ , this processing will enhance the performance of the entire system since the class centroid will move small distances to include some misclassified documents in addition to the original correctly classified ones. This leads to better results than found with traditional approaches.

### **Example:**

Suppose we have the traditional centroid classifier with its drawbacks, as shown in Fig. 3.1. To explain how our proposed approach addresses the drawbacks of the traditional model, let us consider (threshold  $a$  for class A) and (threshold  $b$  for class B) after we categorized all the training set of documents. The system then picked out all the misclassified documents between the centroid class A and the threshold  $a$  (such as  $d1a$ ,  $d2a$ ,  $d3a$ ,  $d4a$ ), which were classified to other classes than class A, then, appended these misclassified documents to the centroid class A and updated it as (updated centroid class A). Therefore, the modified centroid will move a small distances to include those misclassified documents ( $d1a$ ,  $d2a$ ,  $d3a$ ,  $d4a$ ), which will be correctly classified documents. Similarly, we updated the centroid of class B by adding the misclassified documents between the centroid class B and threshold  $b$  (such as  $d1b$ ,  $d2b$ ) to update the

centroid of the class, which is called the updated centroid classifier of class B, which gains the documents related to class B. This approach could convert some of the misclassified documents to correctly- classified documents. This supposition resulted in improving the automatic text classification system performance with high accuracy.

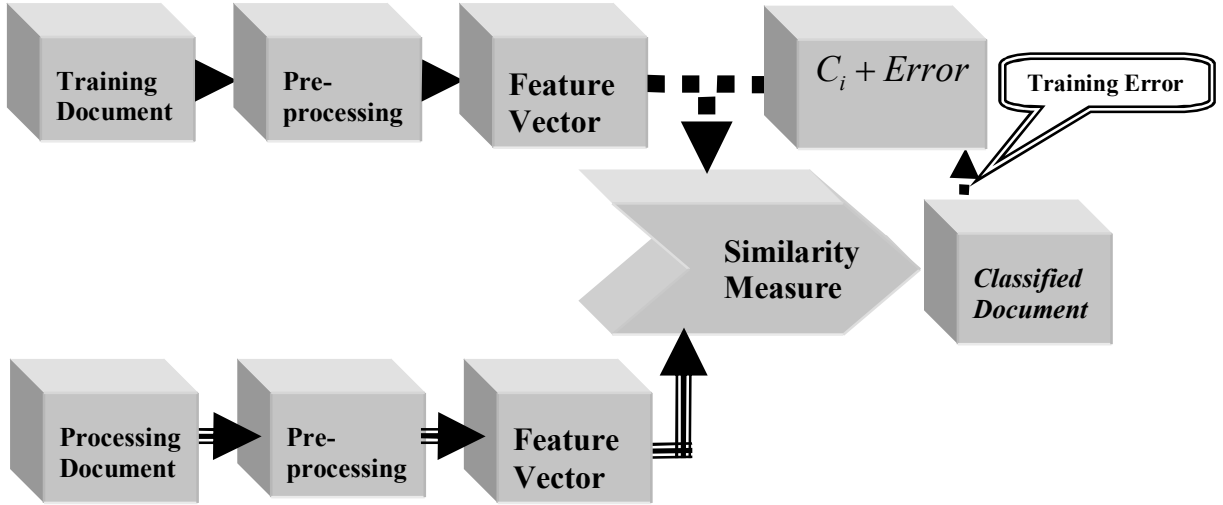


**Fig. 3.1 Modified Centroid Classifier**

### 3.1.1 Modified centroid classifier system outline

Figure 3.2 shows the outline of the new approach. From Fig. 3.2, it is clear that the following steps were carried out with the two main processing methods A and B. In

processing A, compared with the traditional centroid classifier method, the ‘training model’ step is updated with three more new steps, as follows:



**Fig. 3.2 Outlines of the Modified Centroid Classifier System**

### **Processing A: Training the Modifier Classifier**

*Training Documents, PreProcessing, and Feature Vector* are as in traditional method.

#### ***Similarity Measure***

The prototype vector or centroid for each class was created based on formula (2.3). The proposed training system started from this processing. The similarity between the training document vectors and the centroid vector for each class is evaluated by using the dot product between the class vector and the query vector.

### ***Classified Document***

The system classified the training set of documents that were assigned to the class  $c_i$  whose class prototype vector is the most similar to that document described by formula (2.4). This means the accuracy rate should be 100%; however, it is not, so there are some misclassified documents. This is called ***training error***.

### ***C<sub>i</sub> + Error:***

This adds the most similar training errors belonging to a certain class to its centroid and discards the training errors that have low similarities with their class, based on a certain threshold value of 0.2 (after trial and error), according to formula (3.1) in order to get the trained modified centroid classifier.

### **Processing B: Testing the Modified Classifier**

This processing is exactly the same as the processing B procedure in the traditional method except that it replaces the traditional centroid vector with the modified centroid vector for the new approach in order to produce the similarity between the query document vectors and the modified centroid vectors.

Using the new modification, the proposed method could solve the problem of the traditional centroid classifier and represent a more correct classification of the documents.

## **3.2 A New Term Weight**

A new term weight is developed and used to enhance the N-gram word based model. The enhanced N-gram word based model and the centroid classifier model are merged together to construct a hybrid mode for document classification task.

There is useful information for the classification task in the training data set. This information can be used to decide the importance of the term for the classes. To use the powerful of this information we first need to specify the information as follows:

- 1- Term frequency in the class.
- 2- Frequency of documents which contain the term in the class.
- 3- Term distribution in the class.
- 4- Distribution of the documents which contain the term in the class.
- 5- Term frequency in other classes.
- 6- Frequency of documents which contain the term in other classes.
- 7- Term distribution in other classes.
- 8- Distribution of the documents which contain the term in other classes.

Eight formulas are established to extract all the needed information from the training data set. The eight formulas are calculated as follows:

#### **1- Term frequency in the class**

$$TF_{i,k} = \frac{t_{i,k}}{\sum_m t_{m,k}} \quad (3.2)$$

Where  $TF_{i,k}$  is frequency of term  $t_i$  in class  $c_k$ ,  $t_{i,k}$  is the time of occurrence of term  $t_i$  in class  $c_k$  and  $\sum_m t_{m,k}$  is summation of occurrence of all terms in class  $c_k$ .

Formula 3.2 provides a direct relationship between the importance of the term in the class and its frequency. This means that the term  $t_i$  is important for the class  $c_k$  when the term  $t_i$  has a high frequency in the class  $c_k$ .

## 2- Document frequency in the class

$$DF_{i,k} = \frac{d_{i,k}}{d_k} \quad (3.3)$$

Where  $DF_{i,k}$  is frequency of the documents which contain term  $t_i$  in class  $c_k$ ,  $d_{i,k}$  is the number of the documents which contain term  $t_i$  in class  $c_k$  and  $d_k$  is the total number of documents in class  $c_k$ .

Formula 3.3 provides a direct relationship between the importance of the term in the class and the frequency of the documents which contain this term. The importance of the term in a certain class increases as the frequency of documents which contain this term in this class increases.

## 3- Term distribution in the class

$$TS_{i,k} = \frac{t_{i,k}}{\sum_c t_{i,c}} \quad (3.4)$$

Where  $TS_{i,k}$  is distribution of term  $t_i$  in class  $c_k$ , and  $\sum_c t_{i,c}$  is summation of occurrence of term  $t_i$  in all classes  $c$ .

The term which has a high distribution value is an important term for the class. Based on this, formula 3.4 provides a direct relationship between the importance of the term in the class and its distribution.

#### 4- Document distribution in the class

$$DS_{i,k} = \frac{d_{i,k}}{d_{i,c}} \quad (3.5)$$

Where  $DS_{i,k}$  is distribution of the documents which contain term  $t_i$  in class  $c_k$ , and  $d_{i,c}$  is total number of the documents in all classes  $c$  which contain term  $t_i$ .

Achieving a high distribution value for the documents which contain term  $t_i$  in the class  $c_k$  makes term  $t_i$  an important term for class  $c_k$ . Therefore formula 3.5 provides a direct relationship between the importance of the term and distribution of the documents which contain this term.

#### 5- Term frequency in other classes

$$TF_{i,c^*}^* = \frac{\sum_{c^*} t_{i,c^*}}{\sum_{c^*} \sum_m t_{m,c^*}} (nc) \quad (3.6)$$

Where  $TF_{i,c^*}^*$  is frequency of term  $t_i$  in all classes  $c^*$  except class  $c_k$ ,  $\sum_{c^*} t_{i,c^*}$  is sum of occurrence of term  $t_i$  in all classes  $c^*$  except class  $c_k$ ,  $\sum_{c^*} \sum_m t_{m,c^*}$  is sum of occurrence of all terms in all classes  $c^*$  which contain term  $t_i$  except class  $c_k$  and  $nc$  is number of classes which contain term  $t_i$  except class  $c_k$ .

If the frequencies of the term in other classes have a high value, then the importance of the term for the class decreases. Formula 3.6 provides an inverse relationship between the importance of the term for a certain class and its frequency in other classes.



## 6- Document frequency in other classes

$$DF_{i,c^*}^* = \frac{d_{i,c^*}}{d_{c^*}}(nc) \quad (3.7)$$

Where  $DF_{i,c^*}^*$  is frequency of documents which contain term  $t_i$  in all classes  $c^*$  except class  $c_k$ ,  $d_{i,c^*}$  is the number of documents which contain term  $t_i$  in all classes  $c^*$  except class  $c_k$ , and  $d_{c^*}$  is total number of the documents in all classes  $c^*$  except class  $c_k$ .

The importance of the term for the class is decreased when the frequencies of the documents which contain the term in other classes increase. Formula 3.7 provides an inverse relationship between the importance of the term for a certain class and the frequency of documents which contain this term in all classes  $c^*$  except class  $c_k$ .

## 7- Term distribution in other classes.

$$TS_{i,c^*}^* = \frac{\sum_{c^*} t_{i,c^*}}{\sum_c t_{i,c}}(nc) \quad (3.8)$$

Where  $TS_{i,c^*}^*$  is distribution of term  $t_i$  in all classes  $c^*$  except class  $c_k$ , and  $\sum_c t_{i,c}$  is sum of occurrence of term  $t_i$  in all classes  $c$ . Formula 3.8 provides an inverse relationship between the importance of the term for a certain class and the distribution of it in all classes  $c^*$  except class  $c_k$ .

## 8- Document distribution in other classes.

$$DS_{i,c^*}^* = \frac{d_{i,c^*}}{d_{i,c}}(nc) \quad (3.9)$$

Where  $DS_{i,c}^*$  is distribution of documents which contain term  $t_i$  in all classes  $c^*$  except class  $c_k$ , and  $d_{i,c}$  is total number of documents which contain term  $t_i$  in all classes  $c$ . Formula 3.9 provides an inverse relationship between the importance of the term for a certain class and the document distribution in other classes.

The previous eight formulas specify the important terms for each class. Let's find the relationship between the term priority (important) and each value. It is clear that the first four formulas have positive relationship (positive correlation) with the term priority (important). The importance of the term for a certain class increases as the frequency of the term in the class increases. When the number of the documents that contain a certain term in a certain class increases, the distribution of the term in a certain class increases, the distribution of the documents in a certain class increases and the importance of the term increases.

AS for the last four formulas, they have inverse relationship with the term priority (important). The importance of the term for a certain class increases when the frequency of the term in other classes decreases, the number of the documents which contain the term in other classes decreases, the distribution of the term in other classes decreases and the distribution of documents in other classes decreases.

Based on the previous relations, we may merge all the previous formulas in one equation. The proposed merging equation is as follows:

$$tfsc, dfsc_{i,k} = \frac{TF_{i,k} + DF_{i,k} + TS_{i,k} + DS_{i,k}}{TF_{i,c^*}^* + DF_{i,c^*}^* + TS_{i,c^*}^* + DS_{i,c^*}^*} \quad (3.10)$$

Where  $tfsc, dfsc_{i,k}$  is frequency and distribution of term  $t_i$  in the class  $c_k$ , frequency and distribution of the documents which contain term  $t_i$  in the class  $c_k$ .

We exploit formula 3.10 to rank the terms in the class and document profiles. The coming experimentations show the effectiveness of the new term weight technique in classification performance.

### 3.3 Enhancing N-gram Word Based Model

The modified N-gram model uses equation 18 to rank the words or terms inside the class profile. Similarly the query document terms are ranked according to  $tfsc, dfsc$  score. In the proposed enhanced N-gram model, each class has a different profile based on  $tfsc, dfsc$ . The test documents or query documents have different  $tfsc, dfsc$  based profiles. As mentioned before the distance similarity measure between each word in the query document profile and its counterpart in the class profile is calculated based on Manhattan distance in N-gram model.

A new distance similarity measure is applied for the modified N-gram model to calculate the distance similarity. This new distance similarity measure solves the problem of the difference in profiles lengths among the classes and between the document and the class. The new distance similarity measure calculates the distance between the query document profile and the class profile, as follows:

$$\text{distance}(d, c) = \sum_c |\log(c / d)| \quad (3.11)$$

Where  $d$  is the sorted rank of the term  $t_i$  in the query document profile  $d$ , and  $c$  is the sorted rank of the term  $t_i$  in the class profile  $c_k$ .

The new distance measure algorithm solves the problem of difference in profile lengths among the classes and between the document and the class. The new distance formula helps the model to classify the documents accurately. Empirically, the longest class profile contains 4451 words, and the shortest class profile contains 2714 words. That means there is high difference among classes. In this case, the Manhattan distance will be  $(4451-2714=1737)$ , which has bad effect on the classification task. However, when we use the new distance formula, the highest length difference will be  $(\log 4451 - \log 2714 = 0.215)$ . Therefore, the new distance measure will have a good effect on the performance of the modified N-gram. Taking the modulus of  $\log (c/d)$  makes the distance value ranges from 0 to 3.7, which solves the problem of the distinction between the query document profile length and the class profile length. Therefore, formula 2.5 is modified as follows:

$$c_i = \arg \min_{c \in C} \text{distance}(d, c) \quad (3.12)$$

### 3.4 The Hybrid Model Based Center Profile Vector

The new proposed Center Profile Vector (CPV) model based on modified N-gram and Centroid classifier models is exploited to represent the query documents and classes as profile vectors. The modified N-gram, Naive Bays, Support Vector Machine and Centroid classifier models are used as baseline methods. Any classification model has a training step and testing step. The training step and testing step of the CPV model are as follows:

### 3.4.1 The CPV Model Training Step

Firstly, the center vector of the class is calculated as the same as the class prototype vector in the centroid model using formulas (2.1, 2.2 or 2.3) based on the *tfidf*. The CPV model is used to represent each class by the profile vector. This profile vector contains three values. The first value is the center vector of the class using formulas (2.1, 2.2 or 2.3) based on the *tfidf*. The second value is the word itself. The third value is the sorting of the words ranked based on *tfsc,dfsc* value according to formula 3.10. Now the center profile vector represents the class by the previous three values, (the word or term, the *tfidf* of the term and the *tfsc,dfsc* value for term ranking).

### 3.4.2 The CPV Model Testing Step

As same as the class the testing document is represented by a profile contains three vector values, word itself, *tfidf* of word and the rank of term sorting based on *tfsc,dfsc*. In the new model the similarity measure is calculated as follows:

$$c_m = \arg \max_{c_n \in C} \text{sim}(d_t, c_n) = \arg \max_{c_n \in C} \left( \sum_{c_n} \frac{d_{i,t} \cdot c_{i,n}}{|\log((cs_{i,n} / ds_{i,t}) + 0.0001) * NC_i|} \right) * \left( \frac{dc_{i,n}}{dn_t} \right) \quad (3.13)$$

Where  $d_{i,t}$  is the *tfidf* of the term  $t_i$  in the test document  $d_t$ ,  $c_{i,n}$  is the *tfidf* of the term  $t_i$  in class  $c_n$ ,  $cs_{i,n}$  is the rank of the term  $t_i$  in class  $c_n$ ,  $ds_{i,t}$  is the rank of the term  $t_i$  in the test document  $d_t$ ,  $NC_i$  is the total number of the classes which contain term  $t_i$ ,  $dc_{i,n}$  is the number of terms in the test document  $d_t$  which have matching with class  $c_n$ , and  $dn_t$  is the total number of terms in the test document  $d_t$ .

Using formula 3.13, the proposed CPV model will gain the benefits of the centroid model and the modified N-gram model to achieve high classification rate as will be shown in the experimental result.

# CHAPTER 4

## EVALUATION AND EXPERIMENTAL RESULTS

### 4.1 Modified Centroid Classifier

In our study, the Reuters-21578 collection has been exploited as training and testing data. It is divided into 22 files; each of the first 21 files (reut2-000.sgm through reut2-020.sgm) contains 1000 documents, while the last (reut2-021.sgm) contains only 578 documents.

This corpus was collected from Reuters in 1987 and has 21,578 newswire stories related to financial categories. These categories have been manually classified into 135 different sub-categories with relationships to each other. The experimental design in this paper uses the subset of this corpus that consists of the ten most frequent categories, as cited in Table 2. Half of the documents in each class have been randomly selected as training data where the other half has been used as testing data. The description of the used data is as in Table 1:

**Table 1. The training and testing data**

Class	# Documents	Class	# Documents
earn	3964	trade	488
acquisitions	1829	interest	478
money-fx	717	ship	286
grain	582	wheat	283
crude	578	corn	238

Precision-recall and F-measure might be used as evaluation metrics. However, to achieve low misclassification rates and a high degree of separation between different classes on a test set, our experiment found that the accuracy performance of our approach is better than other traditional methods, as shown in Table 3 and Fig. 4.2.

The accuracy is defined as follows:

$$Accuracy = \frac{\# of true positives + \# of true negative}{\# of true positives + \# of true negative + \# of false positives + \# of false negative}$$

### 4.1.1 Centroid Classifier Accuracy

We have utilized the traditional centroid classifier, as described in section 2.1, based on formula 2.3. Then, the testing documents are classified based on formula 2.4. Table 2 shows the document classification accuracy associated with the traditional centroid classifier. From Table 2, the accuracy average of the traditional centroid classifier is 91.5%.

**Table 2 Document Classification Accuracy using Traditional Centroid Classifier**

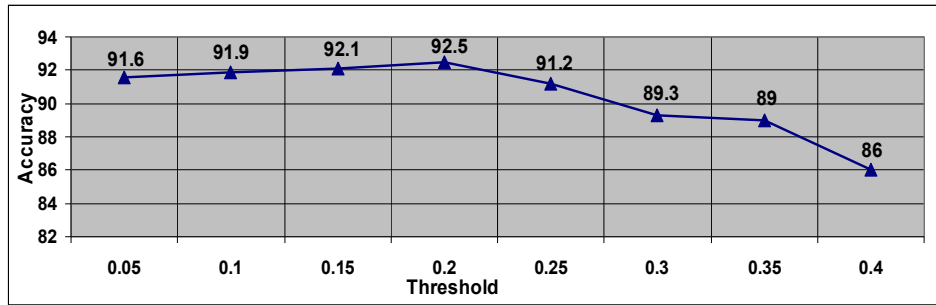
Category	Acquisition	Corn	Crude	Earn	Grain
Accuracy	98.5%	80.6%	81.1%	91.7%	46.6%
Category	Interest	Money-fx	Ship	Trade	Wheat
Accuracy	88.3%	70.8%	78.4%	93.4%	89.0%
Total	Acquisition+Corn+Crude+Earn+Grain+Interest+Money-fx+Ship+Trade+Wheat				
Accuracy	91.5%				

### 4.1.2 Modified Centroid Classifier Accuracy

To overcome the drawback of the previously mentioned approach, we exploit the modified centroid classifier model. In the proposed approach, the threshold value is



chosen based on many trials and errors, using the training set of documents with the following criteria: the suitable threshold is the value that guarantees the highest accuracy of performance. Fig. 4.1 shows the change of accuracy with the threshold. From Fig. 4.1, after achieving the high experimental results of the modified centroid classifier (which gives better document classification accuracy), the chosen threshold value is 0.2.



**Fig. 4.1 Changing accuracy with threshold values**

Table 3 shows the document classification accuracy when the system is using formula 3.1 with threshold value = 0.20. From Table 3, it is clear that the observation accuracy average using modified centroid classifier for all of the most 10 frequent categories is 92.5%. This experimental result demonstrates that the new approach is effective and practical for centroid classification.

**Table 3. Document Classification Accuracy Using Modified Centroid Classifier**

Category	Acquisition	Corn	Crude	Earn	Grain
Accuracy	95.5%	80.6%	84.0%	96.4%	46.7%
Category	Interest	Money-fx	Ship	Trade	Wheat
Accuracy	87.9%	70.3%	77.6%	93.4%	89.0%
Total	Acquisition+Corn+Crude+Earn+Grain+Interest+Money-fx+Ship+Trade+Wheat				
Accuracy	92.5%				

The values of the total accuracies for 91.5% and 92.5% are calculated using the micro average of accuracy of the whole sample data. Table 4 shows the statistical

significance for the total performance of the Traditional Centroid Classifier and Modified Centroid Classifier.

**Table 4. Statistical Significance for Total Performance of Traditional Centroid Classifier and Modified Centroid Classifier.**

The approach	Total accuracy	95% Confidence Interval
Classification Accuracy using Traditional Centroid Classifier	91.5%	90.94%, 92.06%
Classification Accuracy using Modified Centroid Classifier	92.5%	91.97%, 93.03%

## **4.2 Accuracy Comparison for Proposed and Traditional Approaches**

This section describes the previous work on text categorization where the Reuters-21578 collection has been used to evaluate the methods.

Traditional methods (Naïve Bayes, Bayes Nets, Decision Tree, and Support Vector Machine) were utilized by other researchers [44] using the ten most frequent categories from Reuters-21578 collection as training and testing data. Approximately 75% of this corpus is used to build classifiers and the remaining 25% are used as test data in regard to the accuracy of the resulting models in reproducing the manual category assignments. Our approach used 50% as training data and the other 50% as testing data, which establishes that our method is more practical than the traditional one.

From Fig. 4.2, it is clear that the most accurate method is our proposed approach with an average of 92.5% for the 10 most frequent categories, which is higher by 1%, 1.2%, 4.1%, 7.5% and 11% than the traditional centroid classifiers, Support Vector Machines, Decision Trees, Bayes Nets and N Bayes, respectively.

This means that our approach is superior to traditional approaches and enhances the automatic text classification system performance, resulting in the lower cost of time for training and testing processes because this model is a linear model. Moreover, the high classification rate is better than other models, especially with Reuters-21578 collection, which is a noisy corpus with overlapping categories.

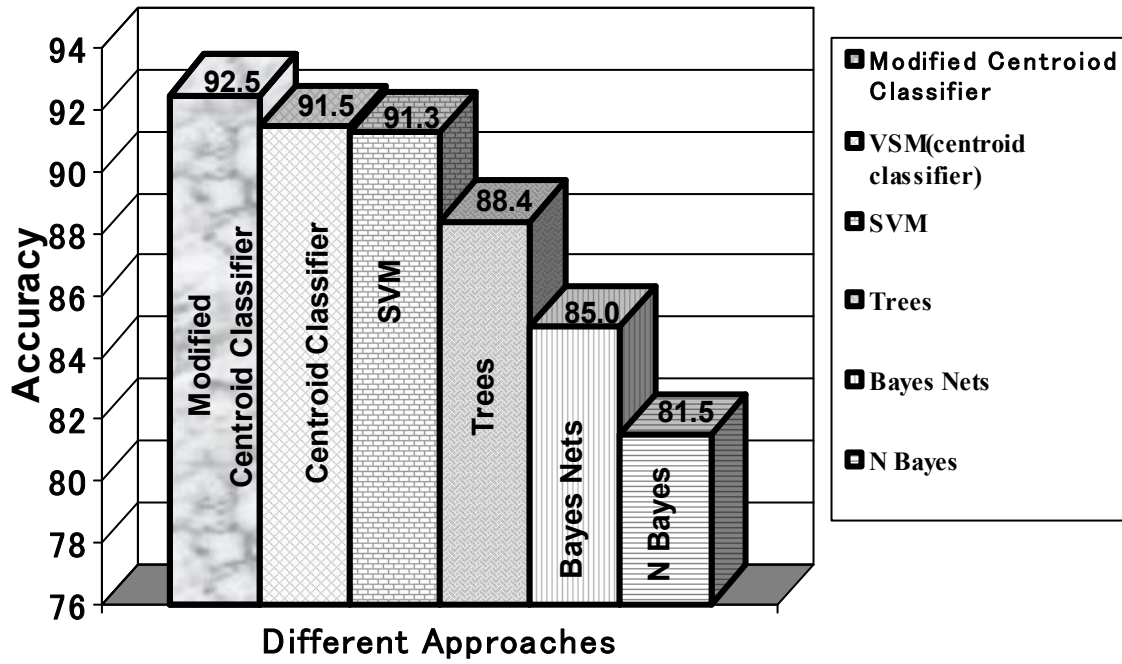


Fig. 4.2 Comparison among Different Approaches based on Reuter-21578 Corpus

### 4.3 The Hybrid Model and Modified N-gram Model

This part of our study exploits the 20 Newsgroups collection as a data set for the experimentations. This corpus contains 18828 documents in 20 different categories which is available in [40]. Half of this corpus has been used for training and other half has been used for testing. The removing of stop words and stemming has been done for all the

documents in the corpus [41]. The F measure has been used to measure the proposed approach performance. F measure can be calculated as follows:

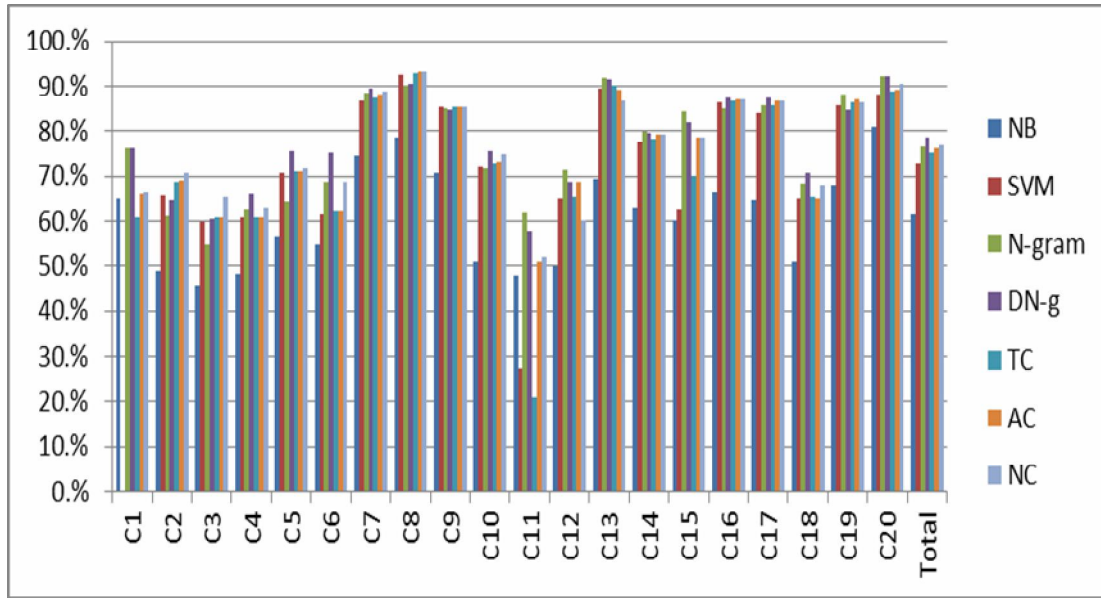
$$F - measure = \frac{2(\text{precision} * \text{recall})}{\text{precision} + \text{recall}} \quad (4.1)$$

Before applying formula 3.13, as baseline models, we have exploited Naive Bays (NB) [65] and Support Vector Machine (SVM) classifiers based on *tfidf* weighting approach. Moreover, we have exploited N-gram (traditional N-gram model with *tfidf* and Manhattan distance), DN-g (N-gram model with *tfidf* and new distance measure), TC (traditional centroid classifier), AC (average centroid classifier) and NC (normalized centroid classifier), as baseline models as well. Table 1 shows the F-measure for each class using traditional NB, SVM, N-gram, DN-g, TC, AC and NC classifiers.

**Table 5: F-measure for Each Class Using Traditional NB, SVM, N-gram, DN-g, TC, AC and NC Classifiers**

Class	NB	SVM	N-gram	DN-g	TC	AC	NC
C1	64.80%	60.95%	76.03%	76.32%	60.95%	66.02%	66.24%
C2	49.03%	68.76%	61.30%	64.59%	68.76%	69.11%	70.57%
C3	45.46%	60.75%	54.65%	60.52%	60.74%	60.79%	65.31%
C4	48.13%	60.91%	62.48%	65.86%	60.91%	60.91%	63.02%
C5	56.29%	71.14%	64.42%	75.55%	71.14%	71.00%	71.71%
C6	54.61%	62.32%	68.86%	75.19%	62.32%	62.23%	68.84%
C7	74.44%	87.49%	88.37%	89.44%	87.48%	88.08%	88.73%
C8	78.66%	92.86%	90.01%	90.38%	92.85%	93.18%	93.16%
C9	70.83%	85.48%	84.98%	84.71%	85.48%	85.51%	85.56%
C10	51.01%	72.94%	71.83%	75.49%	72.93%	73.24%	74.71%
C11	48.00%	20.85%	61.99%	57.91%	20.84%	50.90%	51.91%
C12	49.90%	65.19%	71.30%	68.63%	65.18%	68.75%	60.23%
C13	69.38%	90.29%	91.71%	91.52%	90.28%	89.27%	86.74%
C14	63.04%	78.10%	79.96%	79.73%	78.10%	79.11%	79.10%
C15	60.26%	69.90%	84.46%	81.98%	69.89%	78.70%	78.62%
C16	66.29%	86.83%	84.96%	87.56%	86.82%	87.00%	87.02%
C17	64.76%	85.80%	85.62%	87.38%	85.80%	86.67%	86.89%
C18	51.09%	65.25%	68.51%	70.75%	65.24%	64.93%	67.91%
C19	68.17%	86.33%	88.22%	84.57%	86.33%	87.08%	86.33%
C20	80.83%	88.78%	92.09%	92.28%	88.78%	89.20%	90.49%
<b>Total</b>	61.46%	75%	76.65%	78.63%	75%	76.3%	76.8%

Figure 4.3 shows the F-measure for each class using traditional NB, SVM, N-gram, DN-g, TC, AC and NC classifiers. It is clear from the table 5 and figure 6 that DN-g approach provides the best result over the rest of traditional approaches. The results of the N-gram model in table 5 are low. It might be because the class profiles have the smallest length.



**Fig. 4.3 the F-measure for each class using traditional NB, SVM, N-gram, DN-g, TC, AC and NC classifiers**

Rather than using formula 3.10, in order to investigate the effect of the four term weighting ( $\frac{TF_{i,k}}{TF_{i,c^*}}, \frac{DF_{i,k}}{DF_{i,c^*}}, \frac{TS_{i,k}}{TS_{i,c^*}}, \frac{DS_{i,k}}{DS_{i,c^*}}$ ) on the total system performance, we have exploited **SN-g** (N-gram model with one of the previously mentioned four term weightings rather than *tfsc, dfsc*, and Manhattan distance), **MN-g** (modified N-gram with one of the previously mentioned four term weightings rather than *tfsc, dfsc*, and new distance measure), **CPV&T** (CPV model using class prototype vector of the traditional centroid classifier and one of the previously mentioned four term weightings rather than *tfsc, dfsc*),

**CPV&A** (CPV model using class prototype vector of the average centroid classifier and one of the previously mentioned four term weightings rather than  $tfsc, dfsc$ ), and **CPV&N** (CPV model using class prototype vector of the normalized centroid classifier and one of the previously mentioned four term weightings rather than  $tfsc, dfsc$ ).

Tables 4.3,4.4,4.5 and 4.6 show the proposed approach performance in terms of F measure using the previously mentioned four term weightings rather than  $tfsc, dfsc$  based on **SN-g** (N-gram model with Manhattan distance), **MN-g** (modified N-gram with new distance measure), **CPV&T** (CPV model using class prototype vector of the traditional centroid classifier), **CPV&A** (CPV model using class prototype vector of the average centroid classifier), and **CPV&N** (CPV model using class prototype vector of the normalized centroid classifier).

**Table 6: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting  $\frac{TF_{i,k}}{TF_{i,c}^*}$**

Class	SN-g	MN-g	CPV&T	CPV&A	CPV&N
C1	67.51%	77.96%	75.21%	75.89%	76.26%
C2	52.68%	67.82%	74.79%	74.90%	74.77%
C3	32.43%	62.79%	67.57%	67.63%	70.29%
C4	56.84%	62.24%	66.73%	66.00%	67.78%
C5	39.46%	61.25%	76.73%	76.51%	77.30%
C6	62.08%	70.87%	73.01%	73.06%	74.26%
C7	85.22%	89.40%	91.53%	91.89%	92.17%
C8	78.93%	91.01%	93.81%	93.89%	93.66%
C9	70.72%	86.14%	88.96%	89.18%	89.83%
C10	64.28%	70.32%	73.72%	73.72%	73.23%
C11	52.38%	59.82%	50.79%	59.27%	58.91%
C12	39.59%	72.42%	72.56%	72.49%	68.34%
C13	69.33%	92.04%	92.18%	91.24%	91.18%
C14	70.73%	82.40%	82.13%	82.97%	82.85%
C15	73.90%	83.98%	81.66%	83.81%	82.79%
C16	48.40%	86.49%	88.82%	89.23%	89.23%
C17	51.19%	81.72%	87.04%	87.50%	88.34%
C18	49.29%	56.13%	73.52%	72.67%	72.99%
C19	68.89%	88.70%	90.12%	90.71%	89.90%
C20	86.35%	91.56%	93.92%	94.29%	94.93%
<b>Total</b>	58.26%	76.47%	80.79%	81.04%	81.18%

**Table 7: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting  $\frac{DF_{i,k}}{DF_{i,c}^*}$**

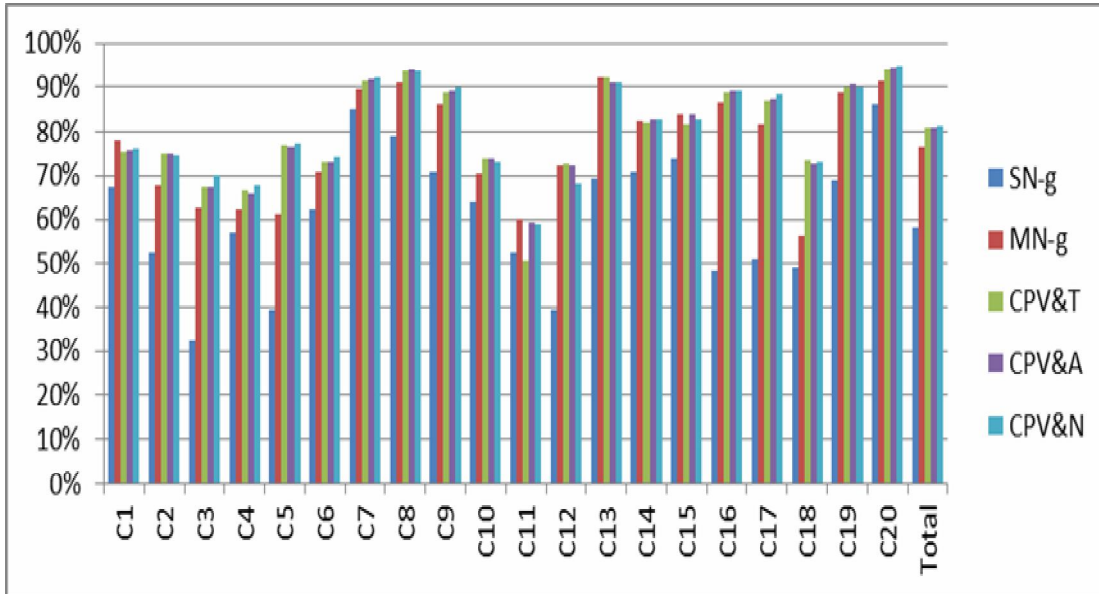
Class	SN-g	MN-g	CPV&T	CPV&A	CPV&N
C1	67.88%	78.12%	75.30%	75.78%	76.27%
C2	43.28%	65.49%	75.08%	75.18%	75.16%
C3	30.94%	60.91%	67.19%	67.19%	70.17%
C4	52.53%	60.28%	67.20%	66.60%	68.12%
C5	34.23%	56.72%	77.71%	77.37%	78.40%
C6	64.73%	70.60%	72.67%	72.83%	74.03%
C7	84.78%	89.38%	91.62%	91.78%	92.61%
C8	79.42%	91.43%	94.20%	94.28%	93.79%
C9	69.85%	85.17%	89.61%	89.43%	89.51%
C10	62.38%	72.06%	75.17%	75.03%	74.69%
C11	51.69%	60.00%	50.99%	58.24%	59.89%
C12	31.74%	72.61%	72.98%	72.59%	68.71%
C13	64.67%	92.14%	92.09%	91.24%	91.26%
C14	70.91%	82.53%	82.01%	82.97%	82.48%
C15	77.13%	84.50%	82.30%	83.96%	83.21%
C16	44.09%	85.81%	88.89%	89.28%	89.03%
C17	49.09%	80.42%	86.92%	87.41%	88.28%
C18	51.64%	56.47%	73.88%	73.12%	73.57%
C19	65.69%	88.43%	90.27%	90.69%	89.92%
C20	85.94%	92.14%	94.04%	94.31%	95.22%
<b>Total</b>	55.69%	75.79%	81.05%	81.20%	81.43%

**Table 8: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting  $\frac{TS_{i,k}}{TS_{i,c}^*}$**

Class	SN-g	MN-g	CPV&T	CPV&A	CPV&N
C1	65.88%	76.55%	75.09%	75.83%	76.05%
C2	36.81%	61.46%	75.08%	75.03%	74.27%
C3	23.88%	58.58%	66.91%	66.79%	70.04%
C4	48.46%	56.12%	66.53%	65.93%	67.45%
C5	29.88%	51.68%	76.07%	75.99%	76.92%
C6	62.09%	70.55%	73.14%	73.22%	75.03%
C7	79.95%	87.91%	92.11%	92.17%	92.59%
C8	70.83%	90.18%	93.96%	93.92%	93.79%
C9	59.59%	83.39%	89.68%	89.46%	89.26%
C10	51.06%	60.58%	73.31%	73.02%	73.09%
C11	47.60%	58.43%	51.43%	58.94%	58.88%
C12	13.08%	72.07%	72.67%	72.09%	68.79%
C13	41.28%	91.15%	92.83%	91.88%	91.63%
C14	55.07%	82.97%	82.96%	83.07%	82.72%
C15	52.63%	82.01%	82.01%	83.65%	82.36%
C16	21.90%	83.45%	89.11%	89.53%	89.32%
C17	29.26%	78.23%	86.69%	87.27%	87.84%
C18	41.15%	51.23%	73.67%	73.25%	73.09%
C19	34.87%	85.65%	90.63%	90.84%	90.00%
C20	80.19%	89.89%	94.23%	94.42%	95.12%
<b>Total</b>	44.34%	72.70%	80.93%	81.01%	81.12%

**Table 9: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting  $\frac{DS_{i,k}}{DS_{i,c}^*}$**

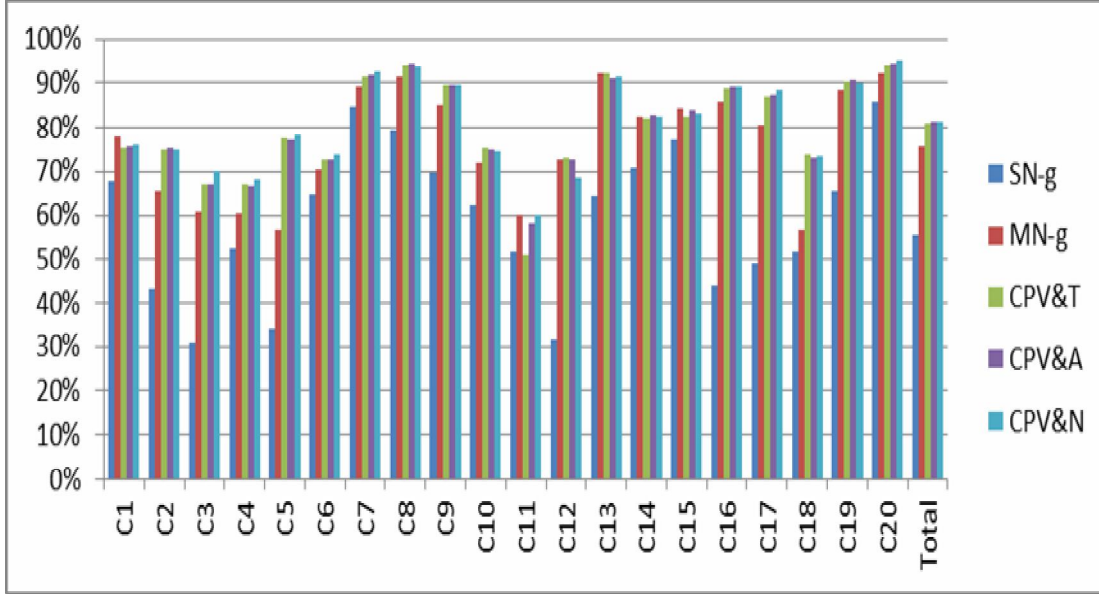
Class	SN-g	MN-g	CPV&T	CPV&A	CPV&N
C1	64.62%	77.35%	75.24%	75.83%	76.31%
C2	26.19%	55.18%	74.84%	74.71%	74.79%
C3	22.39%	57.28%	66.19%	66.25%	70.22%
C4	44.11%	53.77%	66.86%	66.40%	67.84%
C5	27.01%	47.11%	77.15%	76.67%	77.08%
C6	58.16%	68.09%	73.22%	73.22%	75.03%
C7	76.26%	88.24%	92.00%	92.17%	92.49%
C8	69.38%	90.00%	94.14%	94.13%	94.00%
C9	53.97%	81.60%	89.63%	89.80%	89.67%
C10	45.18%	60.34%	73.33%	73.17%	73.60%
C11	46.15%	57.10%	51.87%	58.71%	59.68%
C12	09.88%	70.53%	72.45%	71.91%	68.43%
C13	33.45%	91.23%	92.92%	91.86%	91.71%
C14	47.73%	82.91%	82.68%	83.30%	82.68%
C15	45.68%	81.93%	82.46%	83.73%	82.57%
C16	16.01%	82.00%	89.29%	89.28%	89.03%
C17	25.35%	76.61%	86.66%	87.45%	87.70%
C18	38.69%	50.17%	73.98%	73.56%	73.15%
C19	27.88%	84.27%	90.71%	90.84%	90.18%
C20	77.32%	89.94%	94.14%	94.42%	94.93%
<b>Total</b>	<b>39.55%</b>	<b>71.03%</b>	<b>80.99%</b>	<b>81.07%</b>	<b>81.24%</b>



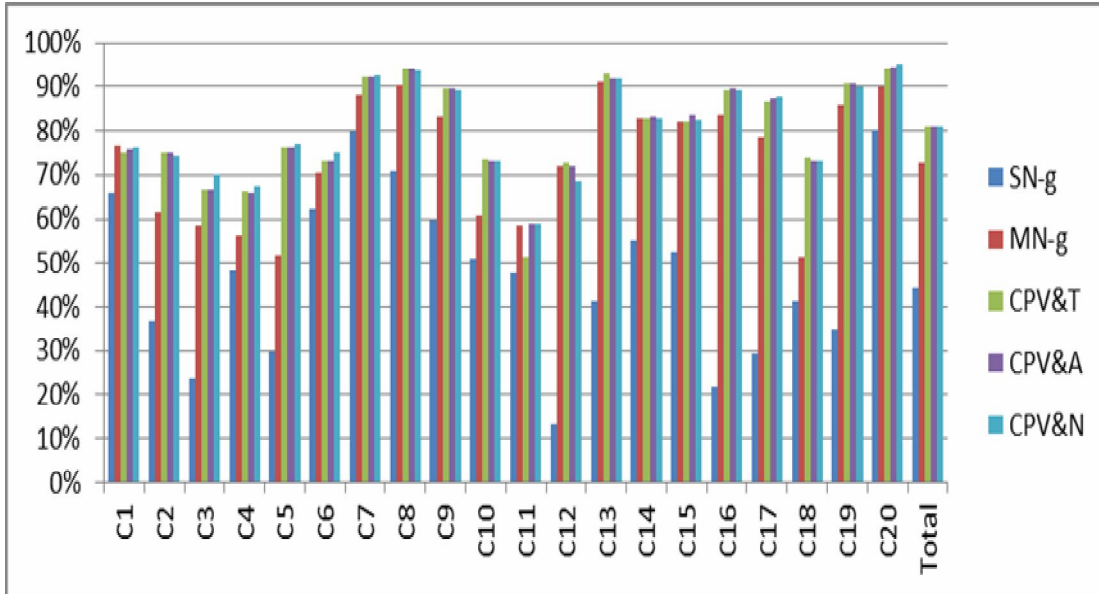
**Fig. 4.4 F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting  $\frac{TF_{i,k}}{TF_{i,c}^*}$**



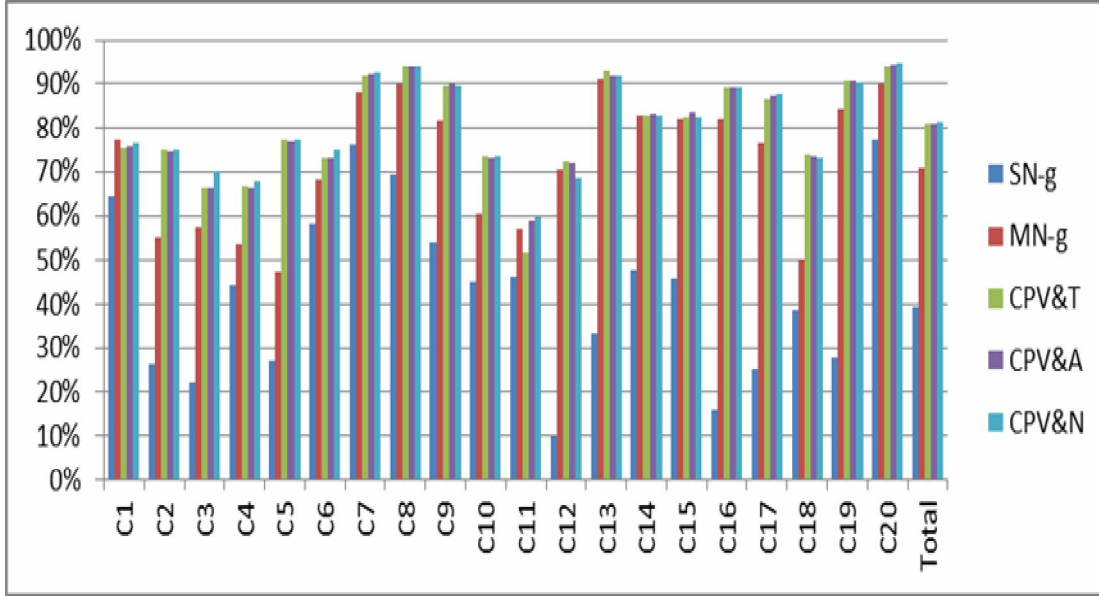
Figures 4.4, 4.5, 4.6 and 4.7 show the proposed approach performance in terms of F measure using the previously mentioned four term weightings rather than  $tfsc, dfsc$ .



**Fig. 4.5 F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N**  
Classifiers based on the term weighting  $\frac{DF_{i,k}}{DF_{i,c}^*}$



**Fig. 4.6 F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N**  
Classifiers based on the term weighting  $\frac{TS_{i,k}}{TS_{i,c}^*}$



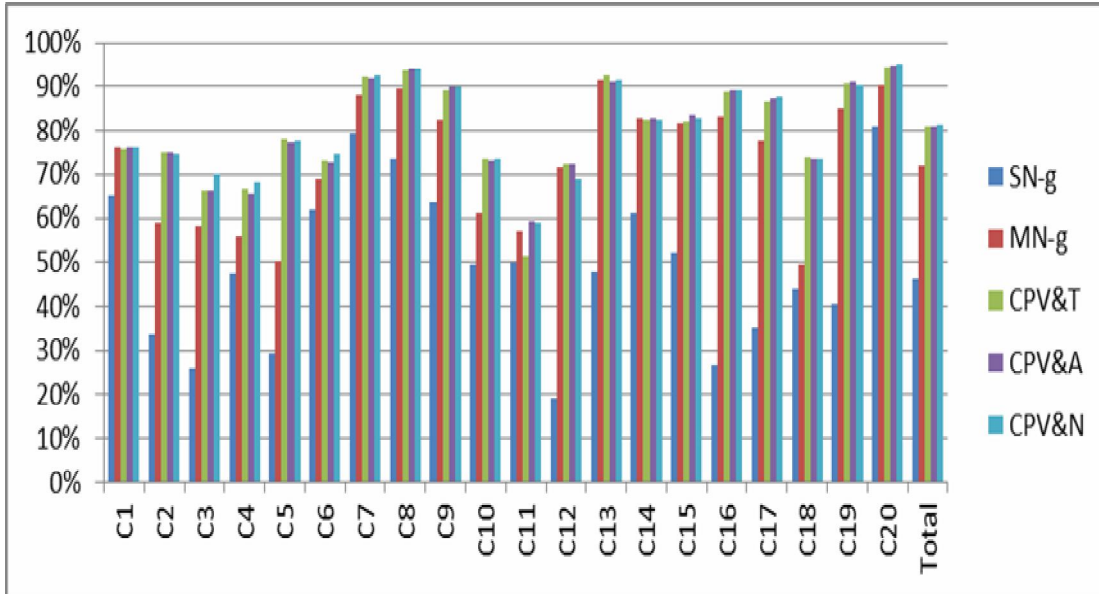
**Fig. 4.7 F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N**  
**Classifiers based on the term weighting  $\frac{DS_{i,k}}{DS_{i,c}^*}$**

It is clear from the previous tables and figures that provide the best results compared with the rest three term weightings.

Repeat the previously mentioned experiments using formula 3.10 to investigate the effect of  $tfsc, dfsc$  on the total system performance. Table 10 and figure 4.8 show the proposed approach performance in terms of F measure using SN-g (N-gram model with  $tfsc, dfsc$  and Manhattan distance), MN-g (modified N-gram with  $tfsc, dfsc$  and new distance measure), CPV&T (CPV model using class prototype vector of the traditional centroid classifier), CPV&A (CPV model using class prototype vector of the average centroid classifier), and CPV&N (CPV model using class prototype vector of the normalized centroid classifier).

**Table 10: F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting  $tfsc, dfsc$**

	SN-g	MN-g	CPV&T	CPV&A	CPV&N
C1	65.15%	75.97%	75.74%	76.17%	76.15%
C2	33.89%	58.87%	74.92%	75%	74.42%
C3	25.99%	58.03%	66.25%	66.25%	70.2%
C4	47.16%	55.93%	66.60%	65.79%	68.18%
C5	29.36%	50.29%	77.90%	77.16%	77.72%
C6	62.05%	68.90%	72.94%	72.86%	74.58%
C7	79.63%	88.11%	92.01%	91.98%	92.69%
C8	73.51%	89.57%	93.73%	94.10%	93.99%
C9	63.66%	82.64%	89.34%	89.81%	89.80%
C10	49.46%	61.19%	73.53%	72.93%	73.51%
C11	49.79%	56.91%	51.5%	59.31%	59.03%
C12	19.33%	71.51%	72.5%	72.19%	68.88%
C13	47.55%	91.28%	92.43%	91.16%	91.32%
C14	61.28%	82.82%	82.51%	82.97%	82.44%
C15	52.00%	81.54%	82.23%	83.65%	82.72%
C16	26.80%	83.3%	88.97%	89.14%	89.21%
C17	35.17%	77.59%	86.57%	87.26%	87.66%
C18	43.92%	49.5%	73.81%	73.31%	73.42%
C19	40.69%	85.05%	90.64%	90.96%	90.31%
C20	81.03%	90.30%	94.32%	94.69%	95.11%
<b>Total</b>	46.00%	71.88%	80.95%	81.02%	81.3 %



**Fig. 4.8 F-measure for Each Class Using SN-g, MN-g, CPV&T, CPV&A and CPV&N Classifiers based on the term weighting  $tfsc, dfsc$**

It is clear from tables 5 to 10 and figures 4.3 to 4.8 that the proposed term weighting ( $\frac{TF_{i,k}}{TF_{i,c^*}^*}$ ,  $\frac{DF_{i,k}}{DF_{i,c^*}^*}$ ,  $\frac{TS_{i,k}}{TS_{i,c^*}^*}$ ,  $\frac{DS_{i,k}}{DS_{i,c^*}^*}$  and  $tfsc,dfsc$ ) provide better results than traditional term weighting ( $tfidf$ ). The best result is still associated with the term weighting  $\frac{DF_{i,k}}{DF_{i,c^*}^*}$  and CPV&N approach. This result reflects the effectiveness of the frequency of the documents which contain term  $t_i$  in class  $c_k$  and the frequency of documents which contain term  $t_i$  in all classes  $c^*$  except class  $c_k$  features. The hybrid model using term weighting  $tfsc,dfsc$  and CPV&N approach provide reasonable result as well which reflects the effectiveness of the new approach.

## **CHAPTER 5**

### **CONCLUSION AND FUTURE WORK**

#### **5.1 Conclusion**

This thesis presents research on automatic text classification, a new approaches for classification field to meet an increasing online information demand. The thesis is divided into 5 chapters: chapter 1 has given the introduction, text classification system and explains the automatic text classification, motivation, problem description, contribution, prerequisite concepts, classification evaluation and outline of the thesis. Chapter 2 has introduced an overview of text pre-process algorithms to build for training and testing data and has described different methods and algorithms for trainable classifier. Chapter 3 has proposed new concrete algorithms for automatic text classification using Centroid Classifier and n-gram. Chapter 4 has described the experimental results on different text features and the result for comparing the baseline method to the proposed methods. Finally, Chapter 5 has focused on conclusion and future works.

We have used the traditional centroid classifier as a baseline model. The proposed approach has used a modified centroid classifier, which has added the most similar training errors belonging to a certain class to its centroid for updating and discarded the training errors that have low similarities with their class, based on a certain threshold value = 0.2. After achieving the high experimental results of the modified centroid

classifier (which has better document classification accuracy), the new approach has chosen a certain threshold value to improve the system's performance.

Our experimental results have found that the modified centroid classifier gives higher classification accuracy than the traditional centroid classifier. This means the modified centroid classifier is an improvement for automatic text classification system performance.

Moreover, we have investigated different approaches for automatic text classification. Firstly, we have exploited the Naive Bays (NB) and Support Vector Machine (SVM) classifiers based on *tfidf* to weigh document's terms. Moreover, we have exploited N-gram (traditional N-gram model with *tfidf* and Manhattan distance), DN-g (N-gram model with *tfidf* and new distance measure), TC (traditional centroid classifier), AC (average centroid classifier) and NC (normalized centroid classifier), as baseline models as well. Secondly, we proposed the modified N-gram model using the proposed *tfsc,dfsc* score to rank the profiles in N-gram. Finally we employed different term weighting schemes to establish the hybrid CPV model based on centroid classifiers and the modified N-gram to improve the classification performance. As shown in the results, the hybrid CPV classification model enhanced the classification performance. The proposed term weighting schemes and the proposed similarity distance measure have good effect on the classification performance for N-gram model.

## **5.2 Future Work**

Further research is needed on the following topics to enhance our system. First of all we have to study other classification systems, including their indexing schemes, methodologies for keeping their taxonomies current, software support, and other tools (almanac, autobiography, bible of book, dictionary, directory, electronic CD-Rom, encyclopedia, front page, handbook, record book, scorebook, thesaurus, year book) for maintenance and searching (Amazon.com, 2001).

Consider the development of an online dictionary or thesaurus for the document classification that would interact with the new taxonomy and assist classifiers and users who work with classification. Such a dictionary would provide not only definitions of terms but also important information about synonymy (related terms), semantic hierarchy (broad terms and narrower terms), and background or common knowledge. Design a new classification system, using a variety of contemporary sources (including the current document classification and the documents mentioned above) that might reflect more accurately the modern structure of the field and its published literature.

We will suggest a mapping function that preserves the historical integrity of the current database and document classification taxonomy in future searching and indexing activities. It should connect the terms in the new taxonomy both with the terms in the current taxonomy and with the terms in the online dictionary.

As we have investigated different approaches for automatic text classification such as: Rocchio's Algorithm, Naïve Bayes Algorithm, Bayes Nets algorithm, K-nearest neighbour algorithm, Decision Tree Algorithm, and Support Vector Machine Algorithm

as a different classification models and compare them with a new concrete centroid classifier. Furthermore, as we have investigated the modified centroid classifier, enhanced n-gram and hybrid CPV model.

In the future work, we will apply the CPV classification model on a large English corpus and other languages such as Arabic, French. Furthermore, we will focus on exploiting other classification models like Decision Trees and KNN to compare it with the proposed CPV model that might improve the automatic text classification performance.

Moreover, we will try to establish a new generation in automatic text classification models by using a smart classification models. In the future work, we think to create a smart model to classify the text. This smart model will update it self automatically. The training phase and testing phase of the smart model will be as follows:

Training step of smart model is as same as the traditional models, but in the smart model, the training step will not stop before the testing step, it will be continued during the testing step without any stop in the future.

Testing step of smart model is as same as the traditional models, but in our smart model, we will add the classified document which has similarity with the class more than 70% to the classification model to update it. That means this classified document will be used as a training document to train the model. Moreover, there are new terms will came with the classified documents. We will add these new terms to the classification model based on some conditions. Like, the number of classified documents which contain the new term should be more than 10 documents in the same class.



# REFERENCES

- [1] Ogura, H., Amano, H., Kondo, M. (2009). Feature selection with a measure of deviations from Poisson in text categorization. *Expert Systems with Applications* 36, 6826–6832.
- [2] Li, T., Zhu, S., & Ogihara, M. (2008). Text categorization via generalized discriminant analysis. *Information Processing & Management*, 44(5), 1684–1697.
- [3] Lertnattee, V., Theeramunkong, T. (2001). Improving centroid-based text classification using termdistribution- based weighting and feature selection. in: *Proceedings of INTECH-01, 2nd International Conference on Intelligent Technologies*, Bangkok, Thailand, 2001, pp. 349–355.
- [4] Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In *ECML-98*.
- [5] Tzeras, K., & Hartmann, S. (1993). Automatic indexing based on Bayesian inference networks. In *SIGIR-93*.
- [6] Lam, W., & Ho., C. (1998). Using a generalized instance set for automatic text categorization. In *SIGIR-98* (pp. 81–89).
- [7] Masand, B., Linoff, G., & Waltz., D. (1992). Classifying news stories using memory based reasoning. In *SIGIR-92* (pp. 59–64).

- [8] Wiener, E. D., Pedersen, J. O., & Weigend, A. S. (1995). A neural network approach to topic spotting. In Proceedings of the 4th annual symposium on document analysis and information retrieval.
- [9] Apte, C., Damerau, F., & Weiss, S. (1998). Text mining with decision rules and decision trees. In Proceedings of the workshop with conference on automated learning and discovery: Learning from text and the web.
- [10] Cohen, W. W., & Singer, Y. (1996). Context-sensitive learning methods for text categorization. In SIGIR-96.
- [11] Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In CIKM.
- [12] Godbole, S., Sarawagi, S., & Chakrabarti, S. (2002). Scaling multi-class support vector machine using inter-class confusion. In Proceedings of the 8th ACM SIGKDD international conference on knowledge discovery and data mining (SIGKDD 2002) (pp. 513–518). New Orleans: ACM Press.
- [13] Joachims, T., W. B. Croft, D. J. Harper, D. H. Kraft & J. Zobel. (2001). A statistical learning model of text classification with support vector machines. In Proceedings of the 24th ACM international conference on research and development in information retrieval (SIGIR'01) (pp. 128–136). New York, USA: ACM Press.
- [14] Nigam, K., Lafferty, J., & McCallum, A. (1999). Using maximum entropy for text classification. In IJCAI-99 workshop on machine learning for information filtering.

- [15] Schapire, R. E., & Singer, Y. (2000). Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3), 135–168.
- [16] Fuhr, N., Hartmann, S., Lustig, G., Schwantner, M., & Tzeras, K. (1991). A rule-based multi-stage indexing system for large subject fields. In *Proceedings of the RIAO'91* (pp. 606–623).
- [17] Schütze, H., Hull, D., & Pedersen, J. O. (1995). A comparison of classifiers and document representations for the routing problem. In *SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 229–237).
- [18] Yang, Y., & Chute, C. G. (1994). An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems*, 12(3), 252–277.
- [19] Apte, C., Damerau, F., & Weiss, S. M. (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3), 233–251.
- [20] Cohen, W. W., & Singer, Y. (1999). Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems*, 17(2), 141–173.
- [21] Lewis, D. D., Schapire, R. E., Callan, J. P., & Papka, R. (1996). Training algorithms for linear text classifiers. In *Proceedings of the 19th international conference on research and development in information retrieval (SIGIR'96)* (pp. 289–297).

- [22] Chakrabarti, S., Roy, S., & Soundalgekar, M. V. (2002). Fast and accurate text classification via multiple linear discriminant projections. In Proceedings of the 28th international conference on very large data bases (VLDB'02). San Francisco, CA: Morgan Kaufmann.
- [23] Aas, K., & Eikvil, L. (1999). Text categorization: a survey. Norwegian Computing Center. Available from <http://citeseer.ist.psu.edu/aas99text.html>.
- [24] Lewis, D. D., & Ringuette, M. (1994). A comparison of two learning algorithms for text classification. In Proceedings of third annual symposium on document analysis and information retrieval (pp. 81–93).
- [25] Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval*, 1(1/2), 67–88.
- [26] Yang, Y., & Liu, X. (1999). A re-examination of text categorization methods. In 22nd annual international SIGIR (pp. 42–49).
- [27] Han, E.-H., Karypis, G. (2000). Centroid-based document classification: analysis and experimental results. in: *Principles of Data Mining and Knowledge Discovery*, 2000, pp. 424–431.
- [28] J.J. Rocchio, Jr. (1971). Relevance feedback in information retrieval. in: G. Salton (Ed.), *The SMART REtrieval System: Experiments in Automatic Document Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1971, pp. 313–323.

- [29] Ittner, D.J., Lewis, D.D., Ahn, D.D. (1995). Text categorization of low quality images. in: Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, USA, 1995, pp. 301–315.
- [30] Joachims, T. (1997). A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. in: D.H. Fisher (Ed.), Proceedings of ICML-97, 14th International Conference on Machine Learning, Nashville, USA, Morgan Kaufmann Publishers, San Francisco, USA, 1997, pp. 143–151.
- [31] Liu, Y., Yang, Y., & Carbonell, J. (2002). Boosting to correct inductive bias in text classification. CIKM, 348–355.
- [32] Tan, S. (2008). An improved centroid classifier for text categorization. Expert Systems with Applications 35 (2008) 279–285
- [33]W. B. Cavnar, and J. M. Trenkle, (1994). “N-Gram Based Text Categorization,” Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval.
- [34] Elmarhumy M., Abdel Fattah M., Ren F., (2009). “Automatic Text Classification using Modified Centroid Classifier”, in the Proceedings of International Conference on Natural Language Processing and Knowledge Engineering, Sep. 24-27, pp. 282-285, Dalian, China.
- [35] Li, X.-B. and Jacob, V.S. (2008) Adaptive data reduction for large-scale transaction data. European Journal of Operational Research, vol. 188, no. 3, pp. 910-924.

- [36] Wilson, D.R. and Martinez, T.R. (2000) Reduction techniques for instance-based learning algorithms. *Machine Learning*, vol. 38, pp. 257-286.
- [37] Pyle, D. (1999) *Data preparation for data mining*. Morgan Kaufmann.
- [38] Dasgupta, A., Drineas, P., Harb, B., Josifovski, V., and Mahoney, M.W. (2007) Feature selection methods for text classification. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 230-239.
- [39] Yang, Y. and Pedersen, J.O. (1997) A comparative study on feature selection in text categorization. *International Conference on Machine Learning*, pp. 412-420.
- [40] <http://qwone.com/~jason/20Newsgroups/>
- [41] Abdel Fattah M., Ren F., Kuroiwa S., (2006). "Stemming to Improve Translation Lexicon Creation from Bitexts", *Information Processing & Management*. Vol.42, No.4, pp.1003-1016.
- [42] L. Lillian, D. Ido and P. Fernando, (1997). "Similarity-Based Methods for Word Sense Disambiguation". *Proceedings of the 35th ACL/8th EACL*, (pp 56—63).
- [43] L. Lillian, (1997). "Similarity-Based Approaches to Natural Language Processing". Ph.D. thesis, Harvard University Technical Report TR-11-97.
- [44] Dumais, S. (1998). "Using SVMs for Text Categorization", In *IEEE Intelligent Systems Magazine, Trends and Controversies*, Marti Hearst, ed., 13(4), July/August.

- [45] C. Lee, G. G. Lee, (2006). Information gain and divergence-based feature selection for machine learning- based text categorization, *Information Processing & Management*, 42(1) 155–165.
- [46] Shankar S, Karypis G. (2000). Weight adjustment schemes for a centroid based classifier, Technical report. Department of Computer Science, University of Minnesota.
- [47] Elmarhoumy M., Abdel Fattah M., Suzuki M., Ren F., (2013). “A New Modified Centroid Classifier Approach for Automatic Text Classification”, *IEEJ TRANSACTIONS ON ELECTRICAL AND ELECTRONIN ENGINEERING*, Volume 8, Issue 4, pp. 364-370.
- [48] Elmarhoumy M., Ren F., (2012). “A New Hybrid Model for Automatic Text Classification”, in *World Congress on Computer Science and Information Technology, WCSIT’12*, December 23-27.
- [49] Elmarhoumy M, Ren F. (2012). “A New Hybrid Model for Automatic Text Classification” in *The Online Journal on Computer Science and Information Technology (OJCSIT)*, Vol. (3)– No. (2), pp. 132-137.
- [50] W. B. Croft and D.J. Harper (1976). “Using probabilistic Models of Information retrieval without Relevance Information”. *Journal of Documentations*. Vol. 35, No. 4, (pp. 285-295).

- [51] F. Fukumoto (1996). "Automatic clustering of articles using dictionary definitions".  
In Proceeding of The 16th International Conference on Computational Linguistic  
(COLINGO'96), (pp.406-411).
- [52] V. Vapnik. The Nature of Statistical Learning Theory, Springer-Verlag, 1995.
- [53] Y. Yang & J.O. Pedersen (1997). A comparative study on feature selection in text  
categorization. In Machine Learning: Proceedings of the Fourteenth International  
Conference (ICML'97), pp.412-420.
- [54] Yang (1998). An evaluation of statistical approaches to text categorization. Journal  
of Information Retrieval. Submitted.
- [55] J. Friedman, T. Hastie and R. Tibshirani, (1998). Additive Logistic Regression: a  
Statistical View of Boosting. Tech. report Stanford University, July23.
- [56] Y. Yang (1999). An evaluation of statistical approaches to text categorization.  
Journal of Information Retrieval, 1(1/2), 67–88.
- [57] G. Salton & M. McGill (1983). Introduction to Modern Information Retrieval.  
McGraw Hill.
- [58] S. T. Dumais, (1991). Improving the retrieval information from external sources,  
Behaviour Research Methods, Instruments and Computers, Vol. 23, No .2, pp.  
229-236.



- [59] E.-S. Atlam, K., Morita, M. Fuketa and Jun-ich Aoe (2006). “Automatic Building of New Field Association Word Candidates Using Search Engine”, Information Processing & Management Journal, Vol.42, No. 4, pp.951-962.
- [60] E.-S. Atlam, M., Fuketa, K., Morita and J. Aoe (2003). “Documents Similarity Measurement using Field Association Terms”, Information Processing & Management, Vol.39, No.6, pp. 809-824.
- [61] C. Buckley, G. Salton, J. Allan and A. Singhal., (1994). “Automatic Query Expansion Using” SMART: TREC 3, In Proc. 3rd Text Retrieval Conference, NIST.
- [62] K. Dohalgren (1982). Naïve semantics for Natural Language Understanding, IBM Corporation, Los Angeles Scientific Center.
- [63] K. Sagara and K. Watanabe (1998). “Extraction of Important Terms that Reflect the Contents of English Contracts”. Journal of Special Interest Groups of Natural Language & Information Processing Society of Japan (SIGNL-IPJS), (pp. 91-98).
- [64] D. D. Lewis (1995). Evaluating and optimizing autonomous text classification systems. In SIGIR '95, pages 246.
- [65] F. Ren, M.G. Sohrab (2013). “Class-indexing-based term weighting for automatic text Classification”. In Information Sciences 236 PP.109–125.